Febbraio/Marzo 1987, Anno II n. 2
Mensile – Lire 6.000

La rivista per utenti di C-64/128 ed Amiga

COMMODORE GAZETTE

INTERVISTA IN ESCLUSIVA AGLI AUTORI DI BARD'S TALE

ISTRUZIONI INEDITE PER L'8502 DEL C-128

TUTTI I SEGRETI DI GEOS

IL CONSUMER ELECTRONICS SHOW DI LAS VEGAS

LISTATI PER C-64 E C-128

I NUOVI COMMODORE AMIGA 500 E 2000

Sped. in abb. post. Gr. III/70

AZIONE INFINITA

ARTICOLI

RUBRICHE

S·D·I
The Screen Burns With Forbidden Passion and Global War!
Amiga™ 512K RAM required
Joystick required
Master Designer Software Presents a CINEMAWARE™ Production S.D.I.
Executive Producers: ROBERT & PHYLLIS JACOB Associate Producer: JOHN CUTTER Directed by KELLYN BEECK
Screenplay by ROBERT JACOB Compatography by SCULPTURED SOFTWARE INC
Original Score Composed by JIM CUOMO Exclusively Distributed by MINDSCAPE

Una pubblicazione **IHT**

Direttore Responsabile Massimiliano M. Lisa

Produzione IHT Technologies s.r.l.
Coordinamento di Redazione Nicolò Fontana Rava
Direzione Artistica Michele Cadrega
Segretaria di Redazione Paola Cinti

Collaborazione Editoriale
Gianluca Frigerio, Alfredo Macchi,
Marco Napoleone,
Gio Cannaviello, Luca Giachino,
Marco Menichelli
Assistente di Programmazione
Sergio Fiorentini
Corrispondenti USA
William S. Freilich: Sezione Sviluppo
Daniela D. Freilich: Coordinamento Generale
Collaborazione Editoriale USA
Louis R. Wallace

Inviato Speciale USA
Matthew Leeds

Ufficio Materiali
Andrea Calicchio

Data Processing
P. Cinti
Impaginazione e Grafica
Antonio Gaviraghi
Fotografia
A.&G.

Direzione, Redazione, Amministrazione
IHT Technologies s.r.l.
Via Monte Napoleone, 9 - 20121 Milano

Fotocomposizione
Proget
Via N. Battaglia, 12 - 20127 Milano
Fotolito
Sele 3
Via Cadorna 49/51 - 20090 Vimodrone
Stampa
Rotolito Lombarda
Via Brescia 53/55 - Cernusco sul Naviglio

Distribuzione
Messaggerie Periodici
V.le Famagosta, 75 - 20142 Milano
Tel. 02/8467545

Pubblicità
IHT Technologies s.r.l.
Via Monte Napoleone, 9 - 20121 Milano
Ufficio Pubblicitario, IHT
Italia ed Estero 02/794181-799492

SEGRETERIA ABBONAMENTI
Linea per registrazione abbonamenti
02/794181-799492

Commodore Gazette
Costo Abbonamenti

Italia:
10 Numeri L. 58.000
12 Numeri L. 69.000
24 Numeri L. 130.000

Estero:
Europa L. 100.000 (10 numeri)
Americhe, Asia... L. 160.000 (10 numeri)
I versamenti devono essere indirizzati a
IHT Technologies s.r.l.
Via Monte Napoleone, 9 - 20121 Milano
mediante emissione di assegno bancario o vaglia postale

Arretrati
Ogni numero arretrato: L. 12.000 (sped. compresa)

Autorizzazione alla Pubblicazione
Tribunale di Milano nr. 623
del 21/12/85
Periodico Mensile
Sped. in abb. post. gr. III/70

Servizio Abbonamenti
Scrivere a:
IHT Technologies
Servizio Abbonati
Via Monte Napoleone, 9 20121 Milano

Associato
alla U.S.P.I.
(Unione Stampa
Periodica Italiana)

NOTE EDITORIALI

Il grave momento di crisi attraversato lo scorso anno dalla Commodore non sembra che un ricordo. Dopo la ventata di licenziamenti che ha attraversato gli Stati Uniti da costa a costa, da West Chester a Los Gatos, gli ingenti fidi chiesti alle banche e la ristrutturazione delle linee produttive, la CBM si presenta oggi come una multinazionale risanata dinanzi alla quale si aprono nuovi orizzonti e prospettive di egemonia non più solo sul mercato degli home computer, ma anche su quello dei PC per tanti anni monopolizzato dalla presenza degli elaboratori della International Business Machines (IBM).

Esaminare il moto che ha spinto l'azienda in momenti anche molto critici risulta difficile. Certo è che, anche mentre la stampa di tutto il mondo già preparava i necrologi per quello che – a sentir loro – «era stato» il gigante del boom dell'home computer, le centinaia di migliaia di individui che rappresentano l'utenza Commodore non si sono fatti influenzare più di tanto. Chi era convinto delle eccezionali potenzialità del neonato Amiga lo ha acquistato curandosi sino ad un certo punto della pubblicità negativa proveniente dalla stampa e della concorrenza dell'Atari capeggiata dall'ex fondatore della Commodore (Jack Tramiel), che ha proposto una sorta di mini-Amiga a basso costo sicuramente in parte scopiazzato dai progetti occhieggiati in casa Commodore prima della sua dipartita.

Il continuo appoggio di una larga fascia di utenti convinti delle capacità tanto dei cavalli di battaglia ad otto bit, quanto del nuovo sedici/trentadue, ha consentito l'evoluzione dell'interessante progetto Amiga 500 e 2000, che colloca oggi la CBM in una posizione di estrema competitività non solo nei confronti di rivali «storici» quali l'Atari e la Apple, ma anche dell'IBM, il contendente di una fascia di mercato in cui la Commodore si è inserita da neofita, ma non per questo con premesse poco incoraggianti.

L'Amiga 500 è senza dubbio il più potente ed il più economico, nel rapporto prezzo/prestazioni, elaboratore mai presentato nel mercato degli home computer. Una macchina sostanzialmente identica all'A1000 ad un prezzo inferiore ai 500 dollari non può essere che destinata non solo ad eguagliare gli ottimi risultati del glorioso C-64 (per il quale ormai si avvicina la pensione), ma a superarli con un buon margine.

L'Amiga 2000, che ad uno sguardo superficiale potrebbe apparire come un A1000 infarcito alla meglio di novità, è in realta una macchina potentissima in grado di rispondere a qualsiasi esigenza possa venire dal mercato dei computer professionali. Le sue capacità di multidos, multiprocessor e multitasking sono un qualcosa di incredibile per il mercato dei PC sino ad oggi dominato da giganti di argilla contenuti in botti di ferro senza alcuna comunicazione tra loro.

Il nuovo elaboratore dischiude nuovi orizzonti per una fascia di utenti che sino ad oggi è stata abituata all'uso di calcolatori per molti versi ermetici che difficilmente si prestano alla multifunzionalità. Non ultimo va considerato anche l'elemento economico che vede l'A2000 come una macchina ben più potente dei suoi diretti concorrenti proposta ad un costo frazionario.

Nell'intraprendere il cammino accanto ai nuovi elaboratori non si può che concludere questo discorso facendo un plauso a Voi, utenti, che, nonostante molta pubblicità negativa, avete saputo scorgere le qualità presenti nelle macchine in Vostro possesso, consentendo così a mamma Commodore di proseguire un cammino che via via è diventato più sicuro. Fatta la Vostra parte ora tocca a lei e le premesse ci sono... M.L.

Indirizzate tutta la corrispondenza per la rivista a:
Commodore Gazette
La posta della Gazette
Via Monte Napoleone, 9
20121 Milano

Il salto di qualità

Innanzi tutto voglio esprimerVi i miei più sentiti complimenti per la rivista che per la sua qualità vale bene le 6000 lire.

Sono un utente del prestigioso, ma (ahimè) vecchio C-64, che in quattro anni mi ha divertito ed incuriosito portandomi in poco tempo dai primi rudimenti di Basic ad una conoscenza più approfondita di tutti i linguaggi per computer (senza esagerare comunque). Adesso però vorrei fare il «salto di qualità» verso un computer di maggiore potenza. Inutile mettere in evidenza che mi sono subito innamorato dell'Amiga per le sue enormi potenzialità. Tuttavia, e per l'assenza di una buona e corretta manualistica sul mercato, e (soprattutto) per la cronica situazione delle mie finanze, rimane per me ancora un bel sogno. Credo comunque che anche a costo di mutui interminabili e di interessi da strozzino riuscirò entro breve ad acquistare l'Amiga e perciò, quasi per il gusto di averlo già, vorrei porvi alcune domande:

1) per un buon conoscitore del Basic del C-64 è totalmente estraneo il linguaggio AmigaBASIC?
2) A quale fascia di utenti è dedicato l'Amiga, aldilà dei buoni propositi espressi dalla Commodore?
3) È sufficiente il manuale fornito con l'Amiga per imparare a programmare questo computer? In caso contrario esistono altre pubblicazioni?

Simon Soft
Firenze

Rispondiamo in ordine alle Sue domande:

1) la conoscenza del Basic del C-64 si può rivelare di grande aiuto, nonostante questo, è opportuno specificare che l'AmigaBASIC è un linguaggio molto più evoluto del V2 e contiene un numero maggiore di comandi e funzioni.
2) Gli aggettivi multiruolo e multiuso propri della definizione di Amiga 1000 hanno finito per creare disorientamento nel pubblico. L'introduzione dei nuovi modelli A500 ed A2000 consente invece di assegnare al primo elaboratore la fascia di mercato «consumer», ossia quella degli home computer, ed al secondo quella dei sistemi professionali.
3) La documentazione fornita a corredo della macchina è sufficiente per un'infarinatura di base. Per approfondire le varie tematiche proposte, saranno tra breve disponibili numerose pubblicazioni della IHT (l'editore di Commodore Gazette, n.d.r.). Il primo titolo ad essere pubblicato sarà «L'AMIGA: immagini, suoni ed animazioni sul Commodore Amiga», seguiranno altri volumi relativi alla programmazione in linguaggio macchina ed all'AmigaDOS.

Il C-128 e la pirateria

Sono possessore di C-128 da più di un anno e da pochi mesi ho anche il drive 1571. Possedendo solo il C2N le mie esperienze erano piuttosto limitate

essendo l'unità a cassette ormai superata. Con l'arrivo del drive ero convinto che mi fosse stato aperto un nuovo mondo di 128, invece niente. I programmi «seri» si contano sulla punta delle dita (in modo 128) e la maggior parte dei giochi non sono che rifacimenti, neanche migliorati, di giochi per il 64 oppure strane combinazioni di programmi che partono dal modo 128 e girano in modo 64.

Fino ad oggi non ho trovato alcun software che mi soddisfacesse pienamente. Quasi tutto il software in mio possesso è frutto di scambi con amici e quello per 128 l'ho comprato «piratato», a 5000 lire per programma. Il prezzo mi sembra assolutamente concorrenziale, visto che, per esempio, Thai boxing è distribuito dalla Lago a 29.000 lire, e le poste italiane sono quello che sono. Qualche volta sono andato alla ricerca di software originale, ma ho dovuto desistere, visto che gli stessi negozi di computer vendono programmi brutalmente copiati, spacciandoli per originali in bei dischetti colorati o cassette con la copertina a colori fotocopiata. Oltre lo sfogo, il motivo per cui ho scritto questa lettera è di porre alcune domande:

1) inserirsi in programmi, lasciandoci magari il proprio nome di battaglia, è legale ai termini del copyrigth?

2) Essendo interessato alla creazione di animazioni, vorrei sapere se esiste un pacchetto software in grado di fornirmi gli strumenti per creare cartoni animati.

3) Ci sono a Roma delle software house serie, dove si possano trovare prodotti originali?

Evitando la routine di complimenti alla rivista, vorrei rimproverare la poca puntualità (non ci potete tenere sulle spine!), ma alla «Gazette» si perdona questo ed altro.

Luke Viola
Roma

Rispondiamo in ordine alle Sue domande:

1) Ogni programma coperto da copyright viene fornito al suo acquirente in «concessione» e non è altresì permesso modificarne od alterarne in alcun modo il contenuto. La situazione si aggrava ulteriormente se il software con registrato un «nome di battaglia» viene distribuito a terzi o rivenduto. Negli Stati Uniti un'azione di questo tipo è considerata criminale ed è punita con ingenti multe e con la reclusione.

2) In modo 128 Le possiamo consigliare Micro Illustrator ed in quello 64 Movie Maker.

3) Non ci risulta che siano presenti in Roma software house rivenditrici di software originale.

Datamanager 128

Sono un biologo di 27 anni da tempo immerso nel settore dell'informatica e della Vostra mirabile rivista. Vi scrivo, tralasciando i più che meritati complimenti, per fare riferimento alla lettera del Sig. A. Bellovino, apparsa sulla rivista di gennaio del 1987. In tale lettera viene menzionato il SUPERBASE 128 come l'unico database disponibile in Italia per il modo 128. Essendo tuttavia possessore di un DATAMANAGER a 80 colonne, pagato L. 132.000 con manuale e perfettamente funzionante, mi chiedo se non esistano altri programmi di questo tipo circolanti in Italia oppure se lo stesso DATAMANAGER possa rappresentare una delle tante copie pirata arrivate anche qui da noi nel Meridione. Devo infatti precisare che il programma di cui sono in possesso, è stato acquistato in un centro autorizzato Commodore a Napoli... e giudicato come originale.

Sempre facendo riferimento alla lettera del Sig. Bellovino, visto lo scarso software prodotto per il C-128, mi chiedo se questo computer potrà sopravvivere come il glorioso 64 oppure dovrà capitolare al fianco dei suoi fratelli minori C-16 e Plus 4.

Infine vorrei sapere se il programma WORD STAR, creato appositamente per il CP/M plus del C-128, è supportabile da qualsiasi manuale presente sul mercato oppure se occorre reperirne uno specifico.

Calzolari Antonio
Avellino

Il miglior modo per controllare se il software in Suo possesso è originale, consiste nel verificare se è accompagnato da istruzioni stampate, da una confezione e se il dischetto è fornito di un'etichetta stampata. Non ci risulta infatti che nessun pirata sia giunto al punto di ristampare l'intero package di un programma.

Sulla base dell'elevato numero di unità C-128 vendute nel mondo, USA compresi (dato importante dal momento che la maggior parte del software è di origine statunitense, n.d.r.), una sparizione dal mercato del C-128 analoga a quella di Plus 4 e C-16 non è nemmeno da prendere in considerazione. A questo proposito non bisogna dimenticare che Plus 4 e C-16 sono stati ritirati dal mercato americano pressochè immediatamente dopo la loro introduzione e che la loro vendita è proseguita solo in Europa.

L'ausilio di un qualsiasi testo dedicato al Word Star non dovrebbe creare nessun problema a chi lo utilizzi nel modo CP/M del C-128.

Il monitor dell'Amiga

Innanzitutto complimenti vivissimi per la meravigliosa rivista sotto tutti gli aspetti che, purtroppo, ho scoperto solo da poco, ma ho già provveduto alla richiesta delle copie arretrate, anche se le ho trovate un poco care al doppio del prezzo di copertina.

Tralasciando questo dettaglio venale, Vi scrivo invece perché, avendo acquistato da poco un Amiga per fare della grafica, ho dovuto constatare purtroppo un lato negativo del monitor e cioè che le immagini quando si è in

alta risoluzione (640x400) tremolano ed a lungo andare la situazione diventa molto fastidiosa e gli occhi si stancano molto, specie per chi ha gli occhiali. A questo proposito volevo chiederVi se non ne esistono con fosfori molto più persistenti in modo da evitare il tremolio, oppure tali monitor se collegati all'Amiga danno lo stesso risultato?

Un'altra cosa che mi sta molto a cuore, e non credo solo a me, è la quasi assoluta mancanza di software in italiano (sic!). Quello esistente è quasi sempre, purtroppo, tutto copiato ed in inglese, per cui si è costretti a passare ore ed ore sul vocabolario per sapere quello che dicono le istruzioni accluse al programma, molto spesso fotocopiate. Ma alle varie S/H costa poi molto far tradurre i vari programmi? Infatti, sono convintissimo che se fossero venduti anche ad un prezzo leggermente maggiore, ma originali ed in italiano nessuno direbbe nulla. Parlando di copie ho notato con vivo piacere la Vs. campagna contro le copie pirata, ed a questo riguardo volevo chiederVi se esiste un sistema abbastanza sicuro per sapere quando un programma è originale oppure copiato. Le copie pirata purtroppo esistono anche per il fatto che l'originale, sempre che sia tale e quando esiste, viene venduto a cifre da capogiro, mentre la copia, garantita funzionante come l'originale, ha un prezzo bassissimo e così quasi tutti scelgono la copia per ovvie ragioni economiche. Dovrebbero esserci invece delle S/H serie che importino i programmi originali, magari traducendoli, per poi venderli a prezzi accessibili, ma fino ad ora non c'è ancora nessuno che io sappia.

Perché non pubblicate un bell'articolo su come si può fare per acquistare direttamente dalle varie S/H degli U.S.A. i programmi originali (infatti vedo dai listini che non costano molto), spiegando bene tutta la prassi burocratica relativa alla richiesta, al pagamento (se in dollari o in lire) e come si fa per eseguire il pagamento, cioè se anticipato o alla ricezione della merce. Credo che la cosa sarebbe interessantissima.

Aldighieri Diego
Mantova

Il problema della vibrazione del display dell'Amiga in alta risoluzione 640x400 sarà risolto dall'introduzione del nuovo monitor Commodore ad alta persistenza A2080, il cui prezzo di listino risulterà essere inferiore ai 500 dollari. Non è ancora stata fissata una data di uscita sul mercato italiano, che non dovrebbe comunque protrarsi oltre il prossimo SMAU di settembre. Per quel che riguarda la distinzione di un programma originale da uno copiato il miglior modo è quello di verificare se è completo di package originale stampato dalla casa madre.

Per ordinare software dagli USA è sufficiente inviare una lettera d'ordine in lingua inglese alla casa madre con allegato un assegno bancario internazionale o la fotocopia di un vaglia postale internazionale. La somma totale deve essere data dal prezzo di listino maggiorato delle spese postali (che spesso indichiamo in queste stesse pagine, n.d.r.). Nel caso non si riesca a reperire l'importo delle spese postali (shipping & handling, n.d.r.) è buona norma maggiorare il prezzo di listino dell'importo forfettario di 10 dollari. Il pagamento si effettua in lire italiane alle poste o alla banca e l'ente accettante provvede al cambio nella divisa statunitense.

MIDI per Amiga

Credo di essere stato fra i primi a mettere da parte il glorioso C-64 e a tentare di domare, almeno un po', l'ultimo nato in casa Commodore (ed a ben vedere, visto come si stanno mettendo le cose, vedi nuove versioni 500/baby=tiro-mancino, ci ho rimesso in dindi... e non sono disposto a perdonarla tanto facilmente a mamma Commodore, ma questa è un'altra storia).

Ebbene ora mi trovo incastrato: ho bisogno di un'interfaccia MIDI (addirittura già vista allo SMAU). Ho quindi telefonato a mammina. Prima mi risponde una segretaria inc..., chissà perché. Poi, pensate un po', parlo coll'impiegato addetto alle informazioni Commodore-Amiga che mi dice «ma cusa l'è 'sto midi, l'è 'na cusa che si mangia?». Va bene, non proprio così ma quasi, comunque dell'interfaccia già annunciata allo Smau, sottolineo, non c'è traccia. Ora io non dubito che fra 5 o 6 secoli, quando il baby Amiga si sarà diffuso come il '64, saranno disponibili per lui tutte le interfacce MIDI possibili... incompatibili con l'Amiga 1000. Ma mi piacerebbe poter usufruire di questa opzione, se non chiedo troppo al magnifico Amiga, fra due o tre mesi o addirittura (bestemmia!) al prossimo numero. A parte gli scherzi vi prego ditemi qualcosa, pensate al mio sintetizzatore Casio CZ101 che sta quasi facendo la muffa in mancanza di questo magico cavetto. Mi basta uno schizzo del collegamento o del circuito (anche inteso come provvisorio o come prototipo), col saldatore me la cavo, oppure il nominativo di un importatore a cui poter telefonare per averlo o l'indirizzo di una qualche associazione o setta o club che, secondo voci di corridoio, sembra custodisca gelosamente l'agognato attrezzo. Insomma, Voi lassù che potete, datemi la soluzione. Di programmi che prevedono il MIDI ne sono pieno: ma senza interfaccia che faccio?

Renato Rolando
Fiano (TO)

La Informatica Italia (C.so Re Umberto 128, 10128 Torino, tel. 011/501647) importa nel nostro Paese l'interfaccia MIDI Soundscape della Mimetics.

Negli USA sono inoltre disponbili numerosi altri prodotti di questo genere: Bright 1 (Micro Engineering), MIDI for Amiga (Skyless), MIDI Gold (Golden Hawk), MIDI-Designs (MIDI-Designs) e Pitchrider 1000 (DOD Electronics). ■

AMERICAN CHALLENGE E AMERICA'S CUP

Due simulatori di vela a confronto

Sono stati di recente pubblicati due nuovi stimolanti programmi dedicati al Commodore 64, che simulano le varie ed emozionanti fasi delle regate veliche: si tratta di «American Challenge», realizzato dalla Mindscape, e di «Official America's Cup», della Electronic Arts. Entrambi i programmi forniscono al neofita precise indicazioni sui principi fondamentali della navigazione a vela e costituiscono una divertente occasione di ripasso per il velista esperto. Considerando queste premesse, ed il fatto che i due programmi sono ben realizzati nella rappresentazione della tecnica di navigazione a vela e di regata, essi non raggiungono un livello di simulazione di pari accuratezza a quello proposto dal Sailing Simulator, realizzato dalla Sailtech per IBM PC. Comunque – tenendo presente il vasto interesse suscitato in tutto il mondo dalla recente America's Cup svoltasi a Fremantle in Australia alla fine di gennaio – certamente tutti, al di là delle differenze d'età e della perizia marinara, gradiranno mettere alla prova la loro competenza, gareggiando con il computer o con gli amici.

La copertina dell'America's Cup presenta una bellissima fotografia a colori del 12 metri

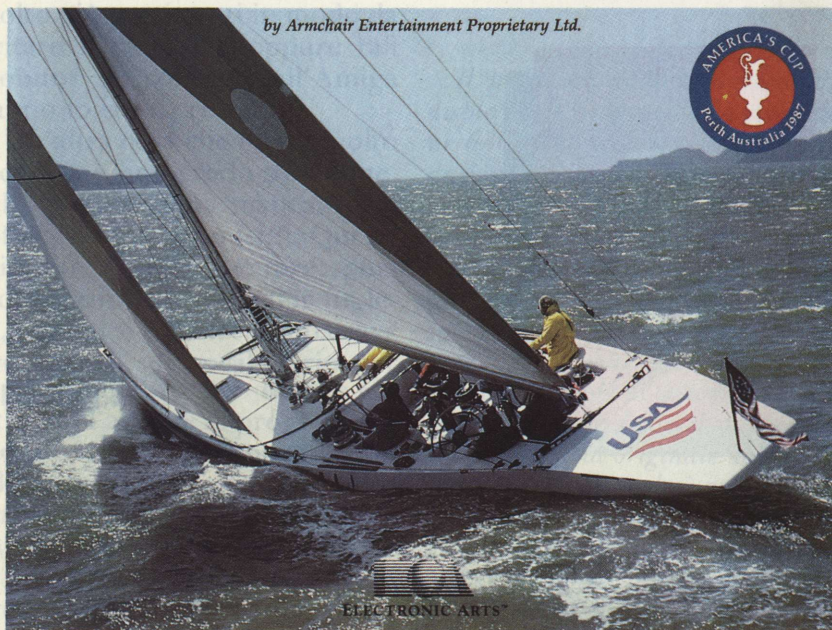
Scheda Critica

Insufficiente.
Un pessimo prodotto che non merita nessuna considerazione.

Mediocre.
Alcuni problemi rilevati in questo programma ci fanno ritenere che ce ne siano di molto migliori.

Discreto.
Lascia lo spazio che trova. Non aspettatevi grandissime emozioni

Buono.
Uno dei migliori programmi della sua categoria.

Ottimo!
Un programma eccezionale che sorpassa tutti gli altri.

In alto: la copertina di America's Cup. In basso: una schermata del gioco

USA, appartenente al St. Francis Yacht Club. L'immagine è stata scattata appena fuori il Golden Gate Bridge, a San Francisco, con lo skipper Tom Blackholler al timone. Sfortunatamente la barca in questione, USA, ha perso 4-0 nei confronti di Stars and Stripes, di San Diego, California, durante le semifinali dello scorso dicembre 1986. USA è stata progettata con l'ausilio del potente e versatile Cary Computer da un gruppo di fisici impegnati nei programmi spaziali della NASA.

Il programma Official America's Cup offre, con il suo percorso di regata triangolare ed il programma di gara comprendente tutti i dodici metri provenienti dalle più diverse parti del mondo, un'eccellente ricostruzione dello svolgimento della competizione. Le norme delle regate, così come il vocabolario dei termini nautici, sono incluse nel programma. A tergo del manuale è anche stampata, per il divertimento degli appassionati, una scheda per il punteggio.

La rappresentazione grafica degli yacht in gara è eccellente. Appena le barche hanno superato la linea di partenza si vede uscire una spirale di fumo dal cannone che, posto sulla barca della giuria, dà avvio alla regata. Le barche sono controllate dal joystick e, utilizzandone una coppia, due «skipper» hanno la possibilità di ingaggiare duelli memorabili. All'inizio della regata il genoa è scelto in base alle condizioni del vento, e può essere in seguito sostituito lungo il percorso appena la velocità del vento accenna a cambiare. Ammainare ed issare il genoa, così come servirsi dello spinnaker appena la barca doppia la boa (per quel tratto di rotta in sottovento), può risultare frustrante e richiede sia perizia che una buona tattica, entrambi requisiti indispensabili nel corso di una regata. Il joystick opera come un verricello: per ammainare il genoa bisogna ruotare il joystick in senso antiorario, per issare la successiva vela scelta, deve essere invece ruotato in senso orario. Se l'angolazione del vento non è esatta lo spinnaker non viene gonfiato dal vento, cosicché diminuisce la velocità della barca. La riduzione della randa può essere controllata con il joystick e bisogna essere molto attenti quando si stramba intorno alla boa, se non altro per il fatto che un uomo può facilmente cadere fuori bordo, con il risultato di perdere 20 lunghezze. «Split screen» sono mostrate non appena una data distanza separa le barche e ciascuno skipper è in grado di selezionare uno scorcio del percorso della regata, la posizione della sua barca, la velocità della stessa, la velocità del vento e la direzione (indicata in un pannello collocato nell'angolo inferiore dello schermo).

Se viene commessa un' infrazione, questa viene immediatamente computata e la relativa penalità consiste nella perdita di quattro lunghezze. Il tempo ufficiale comincia a venir registrato all' inizio della pre-gara intorno alla boa, per poi integrarsi con quello finale. Il manuale, scritto correttamente, mostra al neofita come affrontare molti degli imprevisti che possono presentarsi e costituisce un buon ripasso anche per esperti marinai.

L'American Challenge, realizzato dalla Mindscape, offre uno stimolante ed eccitante panorama di ciò che può significare una regata. Lo skipper controlla la barca servendosi della tastiera per aggiustare le vele e per governare il timone. Il percorso della gara è tracciato nella Sili-

con Bay ed una finestra, collocata nella zona inferiore dello schermo, mostra la direzione della barca, la sua velocità, la velocità del vento e la sua angolazione. Diversamente da «America's Cup» sono presenti i soli comandi per la randa, per cazzarla o mollarla a due velocità, secondo le condizioni del tempo e le tattiche di navigazione. Il controllo del timone è maggiormente vicino alla realtà di quanto non sia quello del sopracitato programma e l'imbarcazione risponde ai comandi con realismo rispetto alle condizioni del vento. È anche inclusa una deriva per veleggiare sotto vento. La simulazione prevede la navigazione solitaria, nella quale si combatte contro il tempo o gli attuali primati, ed un'opzione per due giocatori. Due skipper possono gareggiare uno contro l'altro, inserendo un cavo tra due computer separati, oppure a distanza, collegando il computer ad un modem.

Lo scopo consiste nel riguadagnare l'America's Cup, completando vittoriosamente sette regate preliminari, prima di essere ammessi all'ottava e più diffiicle regata contro la barca australiana detentrice della coppa. La difficoltà delle sette regate preliminari aumenta con il variare dei venti e il sopraggiungere di raffiche o della bonaccia, tutto ciò aggiunto alle normali problematiche che una gara presenta. Per migliorare il tempo di virata della barca intorno alle boe è necessaria molta pratica.

Può essere selezionata anche un'opzione che consente di gareggiare sul medesimo circuito della regata con una barca a motore.

Nella confezione è inclusa anche una cassetta sulla quale è stato registrata una breve guida alla navigazione a vela.

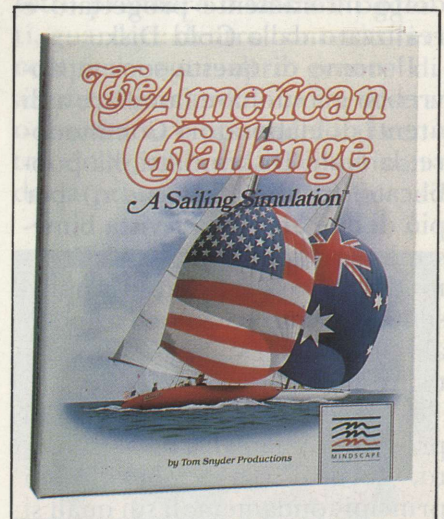
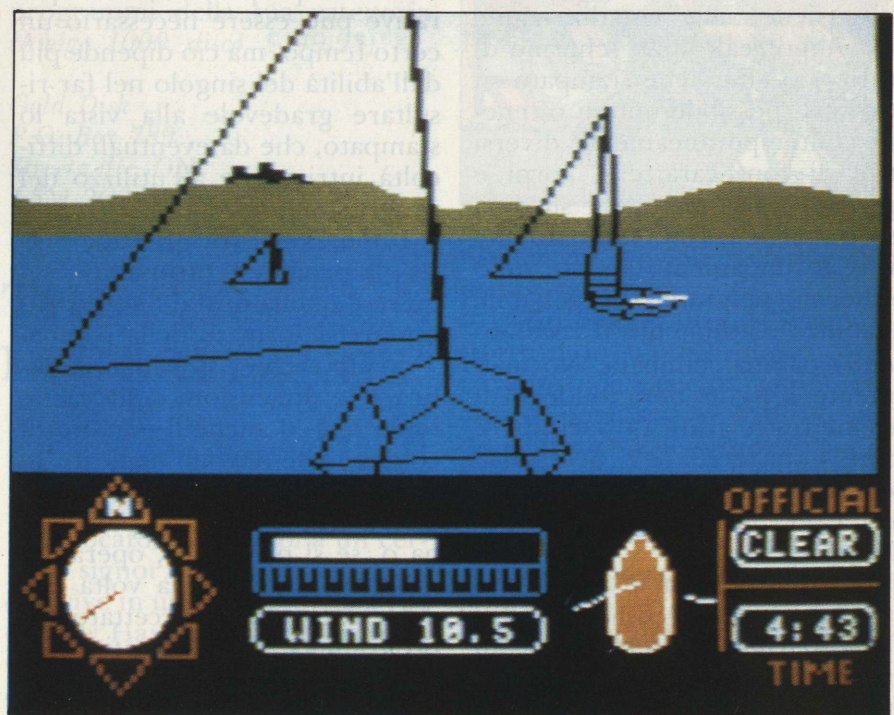
La vista dello skipper può partire da punti diversi, dall'alto per quanto riguarda il percorso di gara, dall'interno e dall'esterno dell'imbarcazione, quando ci si avvicina alla boa, da dritta, da prua o da poppa.

Se avviene una collisione tra due barche durante una gara giocata con un altro computer, sia via modem, che via cavo, la controversia deve essere risolta prima che un'altra gara possa avere inizio. In questo frangente non esiste una giuria ed entrambi gli skipper devono decidere congiutamente quale sia la barca in fallo. Messaggi vari possono essere trasmessi ad un altro skipper durante la gara, permettendo così di metterlo in guardia su una imminente collisione o sui diritti di precedenza.

Il manuale è di buon livello ed include le regole essenziali che i regatanti devono conoscere, un vocabolario di termini nautici ed una bibliografia sufficientemente vasta.

Entrambi i programmi forniscono motivi di interesse per gli appassionati del mondo della vela. Per coloro che sono navigatori provetti è raccomandato comunque tra i due American Challenge, programma che fornisce interessanti tattiche e preziose indicazioni per riguadagnare l'ambita coppa *(The American Challenge: C-64/128 in modo 64, disco, $29.95; America's Cup: C-64/128 in modo 64, disco, $39.95).*

Mindscape Inc.
3444 Dundee Rd.
Northbrook, IL 60062
(001/312/4807667) USA

Electronic Arts
1820 Gateway Drive
San Mateo, CA 944404
(001/415/5717171) USA

In alto: la copertina di America's Challenge. In basso: una schermata del gioco

PAGE SETTER

Finalmente anche il miglior personal presente sul mercato può disporre di un programma dedicato di desktop publishing.

Da tempo si sentiva la mancanza di software di questo genere per Amiga, ed oggi la lacuna è stata colmata. Vi presentiamo infatti Page Setter, un prodotto interamente progettato e realizzato dalla Gold Disk.

Il nome di questa società dovrebbe già suonare familiare agli utenti di elaboratori Commodore; la Gold Disk, infatti, ha pubblicato (negli USA, n.d.r.) per più di due anni una rivista bimestrale su disco dedicata al Commodore 64 e Page Setter rappresenta il primo passo dell'azienda all'interno del mercato del software per Amiga.

È necessario, però, prima di procedere all'analisi del pacchetto, fornire dei ragguagli sui principi fondamentali sui quali si basa la gestione del programma. La componente di maggior rilievo risiede nella visualizzazione in tempo reale sullo schermo di tutto ciò che viene stampato su carta. È possibile quindi ottenere contemporaneamente diversi tipi di fonti-carattere, corpi e grafici, visualizzandoli su una stessa pagina. Dal momento che il testo impaginato appare sullo schermo per intero, è possibile stabilire esattamente il risultato della pagina stampata. Normalmente, se si desidera fondere insieme testo, grafica e vari tipi di fonti-carattere, è necessario combinare l'uso di più programmi. Con l'ausilio del desktop publishing è possibile svolgere funzioni diverse con lo stesso software, ottenendo così un notevole risparmio di tempo ed un maggior controllo sulla qualità finale del prodotto. Page Setter

svolge proprio queste funzioni.

Il programma non è protetto ed è corredato di un manuale molto semplice e chiaro, che fornisce non solo informazioni sull'uso del package in sè, ma presenta anche delle spiegazioni sul desktop publishing in generale. Page Setter è di così facile utilizzo che l'apprendimento delle tecniche di funzionamento del programma avviene in tempi assai brevi. È chiaro che per perfezionare le proprie modalità operative può essere necessario un certo tempo, ma ciò dipende più dall'abilità del singolo nel far risultare gradevole alla vista lo stampato, che da eventuali difficoltà intrinseche all'utilizzo del programma.

Dopo aver caricato Page Setter, la prima operazione da eseguire è quella di stabilire ed impostare il formato della pagina. Tramite menù è possibile modificare le dimensioni della carta, i margini ed anche il numero di colonne che devono apparire in output. Le modifiche possono essere eseguite pagina per pagina o, se si preferisce, operando sull'intero testo. Una volta che il formato è stato accettato dal programma, è consentito all'utente inserire la prima pagina del proprio lavoro.

A questo punto si rende necessario disegnare una gabbia della pagina, che può avere qualsiasi dimensione si desideri ed il cui ingombro si può modificare in qualsiasi momento. Al suo interno vanno inseriti, servendosi di un metodo di immissione rapida come il text editor presente nel programma, o utilizzando un

Turn your pages into a professionally finished image with . . .

PageSetter

Desktop Publishing for the *AMIGA*

Since the desktop publishing boom started in 1982, the industry has exploded. Software, hardware, books and even trade papers have mushroomed to fill a market, which is expected to reach 50 billion dollars world wide by 1990. Not since the advent of the spreadsheet has a single application made such an impact on the computer industry. Large companies, small groups and even individiuals are using their computers to produce crisp, professional flyers, newsletters, signs and resumes quickly and easily, at a fraction of the cost of commercial typesetting. Desktop publishing has brought the power of the press to the people!

PageSetter is a revolutionary new software tool designed to turn your *AMIGA* into a powerful desktop publishing workstation. An 'intuitive' user interface combines ease of operation with complete flexibility of page design. Use magnify to zoom in for detailed work, or pull back for the full page. Combine text (in different fonts and styles) and graphics with complete freedom, and enhance them with a variety of shadings and borders. Spill lengthy articles across

several — linked — boxes

and let **PageSetter** handle the tedious task of formatting. Create stunning artwork and hard-hitting copy quickly and easily using the built in graphic and text editors or import them from your favourite *AMIGA* graphic or word processors. **PageSetter** includes an extensive library of clipart and fonts to allow you to get up and running right away.

And at all times,

"WHAT YOU SEE IS WHAT YOU GET"

Other Features include:

Four justification modes including micro-justification.
Automatic article formatting, through columns and between pages.
Text styles include underlined, [illegible], [illegible], and [illegible].
Graphics may be sized or cropped at will.
Multi-page documents, with variable sizes and formats.
Rulers, grids, columns, and margin indicators.
Measurement in inches or picas.
Prints on any printer in *AMIGA* preferences.

AMIGA & PageSetter What a combination!

The GOLD DISK
P.O. Box 789,
Streetsville, ON
L5M 2C2

PageSetter gives you the Creative Edge!

This document was created with PageSetter

word processor come Textcraft o Scribble, i blocchetti di testo. Per creare grafici ci si serve invece del graphic editor di Page Setter o di altri programmi come DPaint, Graphicraft o Aegis Images.

Si possono creare gabbie di testo differenti nel formato, o per le fonti utilizzate al loro interno, oppure è possibile fondere insieme i vari box in modo che il testo scorra automaticamente dall'uno all'altro.

Il programma fornisce inoltre una serie di opzioni per rendere

l'intero processo di lavorazione meno difficoltoso. Il comando «Page Select» consente di cambiare pagina. Le istruzioni «BoxToFront e BoxToBack» provocano la sovrapposizione delle varie gabbie, consentendo di passare da una all'altra. «Grid» crea una griglia selezionabile tra quattro diverse dimensioni, 1/2 pollice, 1/4 di pollice, 1/6 di pollice o 1/8 di pollice, che aiuta a definire le distanze desiderate. «Ruler» è una sorta di righello al quale possono essere attribuiti diversi parametri di misurazione (1/4 di pollice, 1/6 di pollice o pica).

Un'altra serie di opzioni include «Snap», che allinea automaticamente i riquadri, «Margins», che visualizza i margini impostati per la pagina, e «Boxoutlines», che permette di visualizzare una outline dei riquadri creati. «Magnify» è in grado di visualizzare tre livelli di ingrandimento: il primo è nullo, il secondo visualizza la maggior parte di una pagina standard 8.5"x11" ed il terzo inquadra circa 1/8 di pagina. Con l'ingrandimento maggiore, un pixel sullo schermo equivale ad un pixel sulla stampante, così da poter osservare da vicino i risultati ottenibili. «Page Position» permette di stabilire l'esatta posizione del cursore sulla pagina.

Prendiamo ora in esame i comandi principali del programma: «Box» è utilizzato per generare le gabbie, «QuickText» per l'inserimento veloce di brevi parti di testo (molto utile nell'impostazione di capilettera e similari), «Edit» per modificare il contenuto di una colonna di testo e «Mop» per cancellare una colonna o un grafico. Degna di particolare rilievo è «Link», una delle funzioni più potenti tra tutte quelle disponibili, che, ad esempio, consente, nel caso in cui si sia creato un riquadro contenente del testo, di trasferire attraverso il mouse l'intero blocco in qualsiasi punto della pagina o in un'altra ancora o di miscelarlo con il contenuto di una colonna qualsiasi.

«Unlink» divide i vari riquadri di testo in blocchetti manipolabili singolarmente. «Graphics Adjust», infine, è utile per correggere la posizione delle varie immagini grafiche presenti nella pagina.

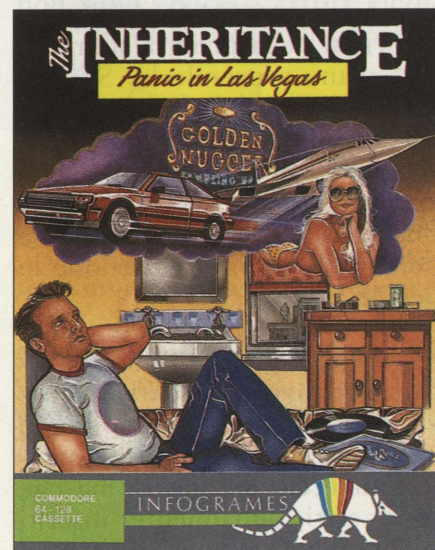
Si possono ottenere anche testi in negativo e quattro diversi tipi di composizione: giustificata, bandiera sinistra, bandiera destra ed epigrafe. L'impostazione di box e fili risulta essere un'operazione estremamente semplice. Esiste anche la possibilità di modificare l'interlettera e l'interlinea e di assegnare spaziature proporzionali, ossia di variare l'interlettera a seconda della morfologia del carattere.

In definitiva, Page Setter è veramente un prodotto completo che contiene tutte le funzioni necessarie per realizzare velocemente l'impaginazione di bollettini e lavori similari. Il più grosso difetto del programma consiste nel fatto che è stato disegnato per fuzionare esclusivamente con stampanti a matrice di punti, penalizzando quello che è l'output classico di software di questo tipo, che è su stampante al laser. Page Setter, comunque, è il primo programma di desktop publishing realizzato per Amiga e, per un prezzo di $149.95, resta in ogni caso uno dei migliori e più competitivi programmi della sua categoria *(Amiga 1000, disco, $149.95).*

Gold Disk
P.O. Box 789
Streetsville, ON
L5M 2C2

THE INHERITANCE

Il giocatore impersona un certo signor Stone, personaggio che vive in uno squallido palazzo di New Haven, nel Connecticut, senza un soldo e assillato dai creditori. Una mattina giunge però un inatteso telegramma. Al signor Stone spetta un' eredità da un fortunatissimo zio arricchitosi col gioco d'azzardo, a condizione però che riesca a vincere un milione di dollari in una sola notte a Las Vegas. Al telegramma sono allegati 200 dollari ed un biglietto aereo.

Il gioco inizia con la squallida camera di Stone vista attraverso i suoi occhi. Un cursore al centro dello schermo, gestito tramite joystick o tastiera, permette di muoversi ed agire, ed è proprio questa una delle caratteristiche più interessanti del programma. Grazie al cursore si possono aprire e chiudere sportelli e cassetti, prendere o lasciare oggetti, guardare dentro contenitori e compiere ogni altro genere di azione. Spostandolo poi a destra o a sinistra, o ponendolo al centro dello schermo, e combinando la pressione del tasto fire, si ottiene un movimento estremamente realistico.

Usciti dalla stanza di Stone ci si trova in un corridoio. Nell'avanzare le varie immagini tridimensionali si avvicinano e ciò dà la perfetta sensazione di un reale movimento. L'obiettivo è quello di uscire dal palazzo in tempo per prendere l'aereo cercando di evitare i numerosi creditori.

A seconda del percorso seguito si trova l'ascensore o la scala. L'ascensore si chiama premendo

il bottone ed una volta entrati si deve indicare il numero del piano desiderato, il tutto attraverso joystick, con notevole rapidità e precisione. L'ascensore può però fermarsi anche ai piani dove è già stato chiamato, nei quali può capitare di incontrare qualche creditore. L'alternativa è quella di scendere le scale molto rapidamente, cercando nel contempo di evitare di soffermarsi sui pianerottoli ove stazionano vicini non troppo amichevoli. Se non si riesce a sfuggire in tempo, infatti, costoro fermano il giocatore chiedendo la restituzione di vari oggetti prestati. In questo caso lo schermo visualizza il creditore nell'azione di parlare e la borsa con gli oggetti trasportati (al massimo 8), il denaro a disposizione ed il tempo passato. Si deve cercare tra gli oggetti (sempre tramite joystick) se c'è quello richiesto. Se non lo si trova il creditore non lascia passare finchè non si è tornati nella propria stanza al 17° piano a prenderlo. Si deve fare una certa attenzione nell' offrire gli oggetti, in quanto, se si prova ad esempio con un arma, c'è chi l'accetta perché l'ha prestata, chi prende Stone per matto e chi per reazione lo uccide.

Altra caratteristica interessante risiede nella possibilità di recarsi, attraverso il cursore, dovunque, anche in quei luoghi che non riguardano direttamente l'intreccio del gioco, ma che sono stati ugualmente inseriti per motivi di completezza e di ricerca del particolare (cfr. telefoni, garage..., n.d.r.).

Giunti al piano terra senza essere incappati in creditori incontentabili ed evitato il guardiano del palazzo, è necessario raggiungere l'aeroporto, dove non mancano nuovi ostacoli di ogni genere.

Nella terza ed ultima parte del gioco si deve far arricchire il signor Stone in un casinò di Las Vegas, giocando d'azzardo alle micidiali macchinette JACKPOT oppure alle carte.

Nel complesso il programma risulta essere un prodotto di buon livello, dotato di una grafica eccezionalmente curata e di un movimento del personaggio molto realistico. Non mancano imprevisti fantasiosi e divertenti che lasciamo scoprire ai nostri lettori *(C-64/128 in modo 64, disco Lit. 25.000, cassetta Lit. 18.000).*

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

WIBSTARS

Mr. Wibstars è un commerciante di software ed hardware. Il suo lavoro consiste nel trasportare con un furgone i prodotti dal grossista ai negozi al dettaglio dove li rivende cercando di trarne profitto.

Il signor Wibstars inizia il suo lavoro giornaliero al magazzino centrale dove deve scegliere quale e quanta merce acquistare: vi sono cassette (L. 5), dischetti (L. 10) e computer (L. 100). Per operare la selezione si deve posizionare un carrello elevatore in direzione del tipo di prodotto desiderato e indicarne la quantità voluta. Naturalmente si è vincolati dal limite del denaro disponibile, che, a inizio gioco, è di L. 200.

Una volta scelti, i prodotti vengono posti su alcuni convogliatori che li lasciano poi cadere: si deve allora fare in modo che entrino nel furgone, impedendo che i pezzi vadano rotti. A carico ultimato lo schermo riproduce una strada vista dall'alto lungo la quale si deve guidare l'automezzo fino ai negozi. Sullo stesso percorso però, è presente anche il furgone di un commerciante concorrente che cerca di creare ostacoli facendo cadere sull'asfalto di varia natura, che potrebbe danneggiare il mezzo del signor Wibstars. Non tutto il materiale caduto è comunque da evitare, in quanto dal furgone concorrente fuoriescono anche prodotti finiti che, se raccolti, possono essere rivenduti con notevole guadagno.

Fin qui il gioco si rivela estremamente banale. Giunti al negozio c'è forse l'unica parte del programma che ha un qualche rilievo. Si deve portare infatti la merce attraverso una serie di convogliatori e nastri trasportatori dal furgone agli uffici. Vi è però un'ascensore, piuttosto difficile da controllare, che rischia di schiacciare i preziosi pacchi da consegnare. Soltanto dopo numerosi tentativi si riesce a recapitare qualche pacco intatto: il che mette in dubbio le possibilità del signor Wibstars di arricchirsi facilmente.

La grafica poco curata e la sezione sonora deludente accompagnano un gioco piuttosto ripetitivo che non è certo una «simulazione» riguardante una società distributrice di programmi come asseriscono le istruzioni: è un giochino come tanti altri che può anche piacere, ma che non emerge certamente dalla mediocrità *(C-64/128 in modo 64, cassetta, Lit. 18.000).*

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

VERA CRUZ

Il giocatore assume le vesti di detective e indaga sull'«affare Vera Cruz» ovvero sul suicidio, che ha forse più dell'omicidio, di una prostituta.

Il gioco è diviso in due parti. Nella prima si svolge il sopralluogo nella casa della defunta. Lo schermo mostra con notevole risoluzione grafica una panoramica della stanza ove si trova il cadavere di Vera Cruz. Naturalmente non si può toccare nulla ed il compito del detective è solo quello di fotografare e cer-

care indizi. Per fare ciò si sposta un cursore sullo schermo sino all'oggetto che interessa e, premuto un tasto, il computer fornisce un ingrandimento dello stesso con le annesse spiegazioni. Sono numerosi gli oggetti che possono rivelarsi utili per le indagini: l'arma, una borsetta, un pacchetto di fiammiferi e uno di sigarette, un'agendina, un portacenere con i suoi mozziconi, una vera e propria indagine. A disposizione del detective c'è il sofisticato sistema informatico denominato DIAMOND COMPUTER NETWORK, che permette di comunicare con gli altri apparati esistenti. Premendo il tasto M (MESSAGE) si può usufruire del sistema, e grazie a codici diversi, ottenere vari servizi. È per esempio possibile contattare un'altra squadra di polizia di una deve risalire da una targa di automobile al suo proprietario, ci si può rivolgere alla Prefettura, oppure, se si desiderano dati sui detenuti, si possono richiedere rapporti alle varie prigioni.

Inizialmente risulta piuttosto difficile per mancanza di sospetti, riuscire a svolgere le indagini, ma, se si persevera con intelligenza, si può cominciare a mettere insieme i dati in proprio possesso per poi proseguire le ricerche su piste diverse.

Oltre al «metodo informatico» vi sono numerose altre opzioni che permettono di usufruire della metodologia d'indagine tradizionale. Si può verificare se certe persone risultano schedate, operare confronti tra i vari alibi o tra certi indizi e i sospetti. Non manca la possibilità di richiedere alla polizia scientifica esami grafologici o delle autopsie.

Si può infine arrestare il sospetto, se ritenuto colpevole, ma non sono concessi errori giudiziari: in questo caso si perde la partita.

Vera Cruz è un gioco tutto sommato interessante, ricco di spunti, realizzato con buona grafica e discreta fantasia. Un adventure che, discostandosi dalle ormai classiche storie ambientate in mondi fantastici, per avvicinarsi alla nostra realtà, si rivela godibilissimo per gli amanti di gialli e polizieschi *(C-64/128 in modo 64, cassetta Lit. 18.000, disco Lit. 25.000).*

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

lettera in cui Vera afferma di essersi uccisa ed altri piccoli, ma importanti particolari. La ricerca degli indizi è fondamentale, perché, una volta conclusa questa fase del gioco, non si può più tornare indietro.

Rientrati in ufficio inizia la seconda parte. Dopo aver redatto un rapporto sugli oggetti che interessano, si può dare il via alla città vicina (il caso si svolge in Francia a Saint Etienne) per conoscere le informazioni, che, contenute nei suoi schedari, dopo breve tempo giungono sotto forma di rapporto. Si può anche contattare il centro informatico di coordinazione di Lione per avere notizie su oggetti o persone (in questo caso viene anche fornita la fotografia). Se si

AGENT ORANGE

Il nome, Agente Arancio, può ricordare il terribile pesticida impiegato dagli Americani nella guerra del Vietnam: per fortuna, invece, in questo caso si trat-

ta di un altro composto ed il suo uso è pacifico. Infatti il giocatore è alla guida di un' astronave del CRAI (Compagnia per le Ricerche sull'Agricoltura Intergalattica), organizzazione creata per fondere le esperienze dell'agricoltura con quelle dei viaggi spaziali al fine di poter colonizzare altri pianeti.

Il più grande nemico del CRAI sono le erbacce che infestano i vari pianeti. Unico rimedio efficace contro di esse è l'Agente Arancio, un potente diserbante, difficilissimo da reperire, dal momento che si trova molto lontano dalla terra.

Scopo del gioco è quello di recarsi su otto pianeti e riempire

l'astronave di diserbante. Per fare ciò si ha a disposizione un'astronave madre (che in realtà serve solo come base appoggio) e otto vascelli minori. Se però si riesce ad ottenere buoni raccolti e ad incassare denaro si possono poi acquistare al mercato delle astronavi mezzi più veloci e meglio armati.

Le astronavi minori contengono dei semi che vanno piantati nel terreno e che una volta cresciuti vanno raccolti. Sullo schermo è visualizzato il peso del raccolto caricato, il numero dei semi rimasti, la posizione, il punteggio ed il numero di vascelli disponibili.

Ad ostacolare il compito vi sono astronavi aliene che devono essere distrutte o evitate. Sono anche da eliminare, sparando a breve distanza, le piante autofertilizzanti sviluppate dalla tecnica genetica aliena.

Il programma non si discosta molto, per concetto strutturale, dai soliti «spara e fuggi», la grafica non è eccezionale ed anche la stessa giocabilità risulta limitata ad un brevissimo periodo di tempo, dopo il quale sopraggiunge la monotonia.

Non si distinguono inoltre molto bene le piante dalle erbacce e dalle coltivazioni aliene, ma questo può essere considerato anche come una difficoltà per rendere un po' meno banale il gioco.

Alcune opzioni introdotte per movimentare l'intreccio (cfr. cambio di pianeta, lancio di astronavi..., n.d.r.) si rivelano del tutto inutili per un gioco di livello complessivamente non certo elevato *(C-64/128 in modo 64, cassetta, Lit. 18.000).*

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

S.D.I.

Mosca, 25 ottobre 2017. Nel giorno del centesimo anniversario della rivoluzione bolscevica, alcuni esponenti dell'esercito sovietico e dei servizi segreti (KGB, n.d.r) occupano tutti gli aeroporti, i centri di trasporto e le stazioni di difesa del Paese, preparandosi a marciare su Mosca per rovesciare il governo.

Viktor Aliyev, uno dei leader della rivolta, è convinto che il nuovo sistema di difesa strategica (Strategic Defense Initiative) degli USA, che sta per essere completato, possa costituire motivo di ricatto nei confronti dei Sovietici e per questo si è impossessato delle basi nucleari e si appresta ad attaccare gli Stati Uniti con i missili a sua disposizione.

In questa atmosfera fantascientifica (o almeno lo speriamo, n.d.r.) è ambientato il nuovo gioco della Master Designer Software, distribuito dalla Mindscape. Sulla scia del successo di Defender of the Crown, la Master Designer ha orientato tutti i suoi sforzi nella produzione del cosiddetto «Cinemaware», ossia di giochi, sia di azione che di avventura, dotati di una grafica così realistica e curata da ricordare da vicino il grande schermo. S.D.I. è sicuramente un gioco meno complesso del precedente Defender, ma non per questo meno avvincente ed interessante. Si tratta infatti di un programma che dedica, rispetto al precedente, molto più spazio all'azione che alla strategia, combinando all'intreccio schermate di grande effetto.

Il giocatore veste i panni di Sloan McCornick, il capitano delle forze orbitali al comando della iniziativa di difesa strategica (SDI) degli Stati Uniti, col compito di affrontare le forze rivoluzionarie in modo da evitare lo scoppio della III° guerra mondiale. Per portare a termine questa missione sono disponibili un caccia stellare ed una stazione orbitale dotata di armi in grado di abbattere i missili sovietici. Non manca l'appoggio del governo sovietico, che ancora si oppone ai ribelli, e, in particolare, di Natalya Kazarian, comandante della stazione spaziale V.I. Lenin.

Scopo del gioco è quello di raggiungere cinque obbiettivi:

1) Distruggere tutti i caccia del KGB.

2) Difendere il proprio suolo dagli attacchi dei missili.

3) Riparare tutti i satelliti SDI danneggiati.

4) Tentare di salvare Natalya quando la sua stazione viene assalita.

5) Dopo aver lasciato la stazione V.I. Lenin, riparare tutti i sa-

telliti ed eliminare i caccia nemici rimasti.

Il gioco

Tutti i comandi di SDI si controllano tramite un joystick, che deve essere inserito nella porta n. 2. Dopo aver caricato il dischetto (direttamente dopo il Kickstart), appare una breve introduzione ed un dispaccio per il capitano McCornick nel quale vengono fornite le prime indicazioni ed istruzioni.

La base spaziale

Terminato il dispaccio la scena si sposta all'interno della base spaziale.

Da qui è possibile accedere al

radar a lungo raggio (scanner) che visualizza una mappa dell'intero globo terrestre e la relativa situazione strategica (schermo a sinistra), controllare lo stato dei satelliti (schermo sulla destra) e servirsi dei satelliti SDI per difendere gli Stati Uniti dagli attacchi missilistici.

Queste operazioni si effettuano portando il cursore sui relativi schermi e premendo il tasto «fire».

Se, invece, il cursore viene posizionato su McCornick (al centro dello schermo), un ascensore lo conduce alla cabina di depressurizzazione e quindi al passaggio per entrare nella cabina del caccia.

Caccia stellare

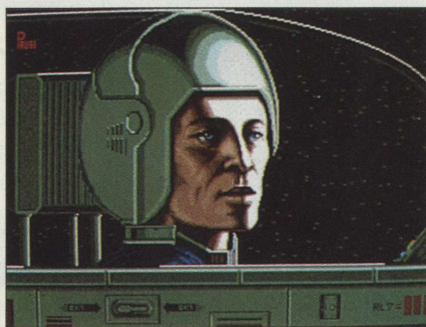
Questo indispensabile aeromobile consente di volare nell'orbita terrestre, di abbattere i caccia sovietici, di riparare i satelliti SDI danneggiati e di tornare alla base spaziale per manovrare il sistema di difesa.

A chi per la prima volta si appresta a cimentarsi con SDI, si consiglia vivamente di allenarsi per un certo periodo di tempo a manovrare il caccia, in quanto è fondamentale essere rapidi e precisi ai comandi, specie nelle situazioni di emergenza.

I movimenti fondamentali dell'astronave sono controllati direttamente tramite joystick. Per aumentare o diminuire la velocità è necessario agire sui tasti «+» e «-».

La plancia del caccia è così configurata: nella parte centrale dello schermo è raffigurata la vista dall'abitacolo (e il radar a lungo raggio «scanner», nel caso in cui venga richiamato), in quella inferiore sinistra è collocato l'indicatore della velocità, al centro il misuratore del livello di carburante (energia) e a destra un segnalatore dell'efficacia degli scudi protettivi. Nella parte

superiore sono presenti, da sinistra verso destra, un riquadro che segnala se c'è qualche messaggio in arrivo, un radar a corto raggio ed un timer relativo ai secondi mancanti all'arrivo delle testate nucleari eventualmente lanciate dai sovietici.

Oltre a combattere i mezzi nemici, il caccia può compiere altre operazioni alle quali si accede tramite tastiera. RICEVIMENTO MESSAGGI: se il led collocato nella parte superiore destra dello schermo lampeggia, indica che si sta per ricevere un messaggio. Premendo il tasto «M» scompare momentaneamente il radar a corto raggio, al posto del quale viene visualizzato il messaggio ed il volto di chi lo invia (Natalya o il luogotenente Andrew Bowman). ATTRACCO: se si desidera attraccare nuovamente alla base spaziale americana o a quella sovietica è necessario premere il tasto «D» e successivamente combinare ai movimenti nelle varie direzioni del joystick, per centrare il ponte di atterraggio, la pressione del tasto «fire», per azionare i retrofreni. Una volta completato l'attracco è possibile fare rifornimento, ricaricare gli scudi e riparare i danni subiti durante i combattimenti. Terminate queste operazioni, si può scegliere se ritornare a combattere (DISENGAGE) o entrare nella base (ENTER AIRLOCK). RIPARAZIONE SATELLITI: i satelliti danneggiati sono quelli visualizzati in rosso sul radar e che sullo schermo principale lampeggiano quando vengono incrociati. Per renderli nuovamente operativi è necessario premere il tasto «R»,

inquadrare il satellite danneggiato e premere il tasto di fuoco. È da notare che quando ci si trova nei modi di attracco o di riparazione non è possibile servirsi dei laser, ciò, quindi, rende l'astronave più vulnerabile. SCANNER: il tasto «S» consente l'attivazione del radar a lungo raggio che visualizza l'intero globo terrestre, la posizione del caccia e dei mezzi nemici.

Difesa dai missili

Durante i combattimenti è posssibile che giunga un messaggio relativo all'avvenuto lancio di missili sovietici. Da quel momento in poi ci sono solo 2 minuti di tempo prima che i missili siano a portata di tiro dell'SDI. Il timer segnala il tempo rimanente ed è necessario trovarsi al-

l'interno della base prima che raggiunga lo zero.

Sistema di difesa

Lo schermo del sistema di difesa è così conformato: nella parte superiore è visualizzato in 3D lo spazio aereo sovrastante gli Stati Uniti, nella parte inferiore un riquadro nel quale compaiono i missili da abbattere.

La procedura per distruggere un missile è caratterizzata da fasi diverse. Dopo aver posizionato il mirino nella parte terminale della traccia della testata ed aver premuto il tasto di fire, nella finestra inferiore appaiono i dati relativi allo stato del satellite primario (in funzione o danneggiato), alla destinazione del missile (nome di una città degli USA), al numero del satellite ed alle coordinate del missile. A questo punto si rende necessario effettuare una scelta, cioè decidere se utilizzare il satellite primario o quello secondario (da selezionarsi necessariamente nel caso il primario sia danneggiato). Per scegliere il primario si deve spingere la leva del joystick verso destra, per il secondario verso sinistra. L'operazione è analoga per scegliere l'uso di un unico fascio laser o di più colpi (FULL BEAM o PULSE).

Una volta effettuate queste operazioni compare il missile, che va inquadrato nel mirino ed abbattuto. Tutte queste operazioni vanno eseguite molto velocemente ed è necessaria una grande abilità e prontezza di riflessi. Prima che un missile tocchi terra, dal momento in cui appare, intercorrono circa 35-40 secondi e gli attacchi sono costituiti da ondate di 6-7 missili che compaiono quasi contemporaneamente.

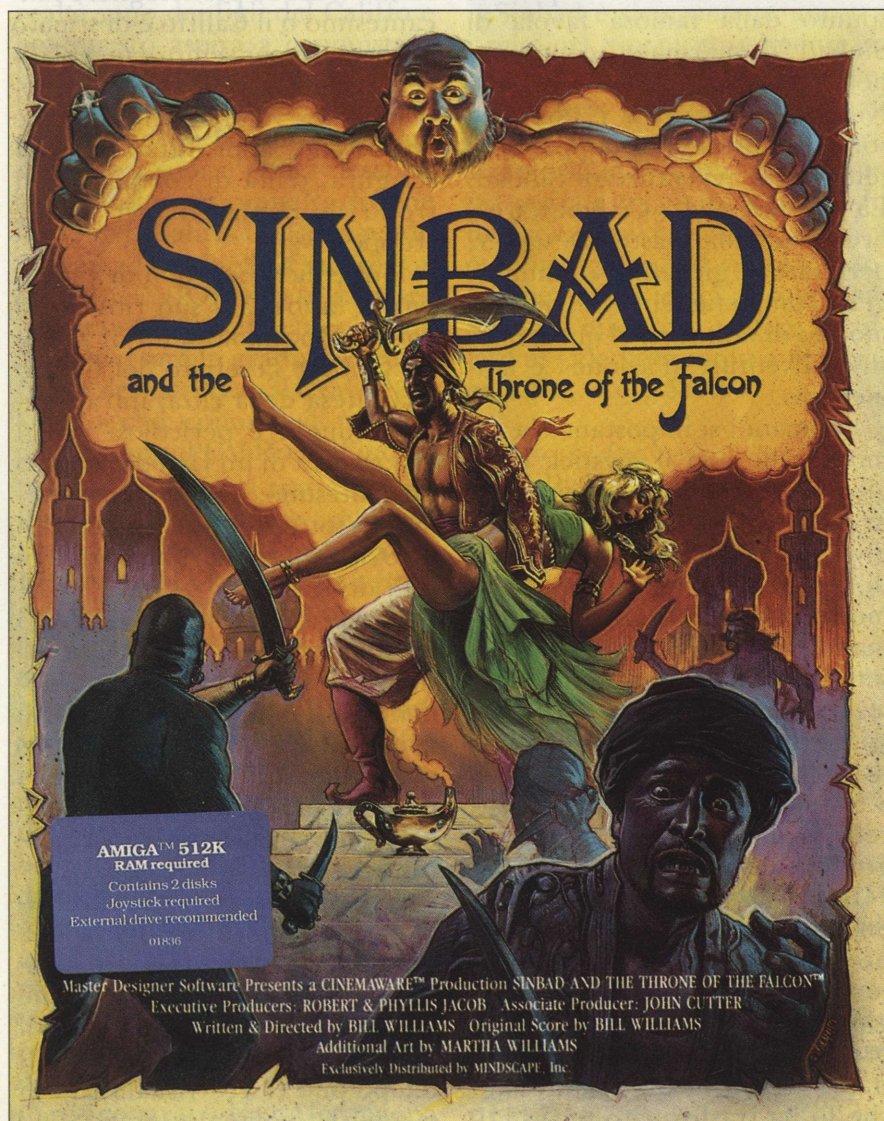
La stazione V.I. Lenin

Quando questa stazione viene attaccata, Natalya, il suo comandante, potrebbe chiedere aiuto. In questo caso è necessario fare almeno un tentativo di salvataggio se si desidera terminare il gioco. Le modalità di attracco al ponte sono analoghe a quelle utilizzate con la base USA. Una volta entrati nella stazione è necessario combattere, con l'ausilio di una pistola laser, contro le guardie nemiche che hanno il 50% di possibilità di mandare a segno i loro colpi (si consiglia di «anticipare» la comparsa del nemico). Un timer collocato nella parte superiore dello schermo segnala il tempo rimasto a disposizione per completare il sanvataggio. La velocità è, quindi, un elemento fondamentale. Non appena si decide di affrontare il rischio e di esporsi al fuoco avversario (ma si può anche eliminare prima tutti i soldati nemici) si deve spostare il mirino sulla parola «RUN», nell'angolo inferiore sinistro dello schermo, e premere il pulsante.

Completata quest'ultima operazione, non resta che lasciare la base sovietica e distruggere i caccia sovietici rimasti per riportare la pace in tutto il mondo *(Amiga 1000, disco, $49.95).*

Mindscape Inc.
3444 Dundee Road
Northbrook, IL 60062
(001/312/4807677) USA

SINBAD E IL TRONO DEL FALCONE

Avventura, azione e strategia si combinano in modo organico all'interno di questo eccezionale gioco, recente creazione

della Master Designer Software.

L'ambientazione prende spunto dalla famosa favola di Sinbad il marinaio, contenuta nella raccolta «The Arabian Nights», e da tutti i film che sono stati prodotti sviluppando il tema delle avventure di Sinbad. La tecnica del «Cinemaware», proposta dalla Master Designer, permette al giocatore di immergersi completamente nel magico mondo di fantasia nel quale questo «adventure d'azione» è ambientato.

I comandi si impostano sia col mouse che con il joystick ed il gioco è strutturato e concepito in modo da risultare «user friendly». Non sono necessarie, quindi, particolari spiegazioni sulle modalità di impostazione dei comandi, mentre è opportuno essere ben informati sui miti, sulle leggende, sui caratteri dei personaggi e dei luoghi, tanti e tali da occupare due interi dischetti da 3.5", necessari a contenere i dati relativi alle numerose schermate grafiche.

Il giocatore veste i panni di Sinbad, il marinaio di Damaron protagonista ed eroe dell'avventura. La principessa Sylphani, figlia del Califfo di Damaron, scopre una mattina che il padre è stato vittima di un incantesimo che lo ha trasformato in un falcone. I medici di corte sono tutti concordi: è necessario trovare subito il modo per sciogliere l'incantesimo o il Califfo è destinato a rimanere per sempre un volatile. Ad aggravare la situazione incombe la figura del Principe Nero, che, approfittando della sciagura, tenta di conquistare il trono di Damaron con le sue truppe.

Molteplici sono quindi i compiti di Sinbad, al suo ritorno in patria dopo molti anni di assenza: guidare l'esercito del Califfo alla difesa della città, intraprendere lunghi e pericolosi viaggi alla ricerca di un rimedio contro l'incantesimo, e tenere costantemente sotto la propria protezione Sylphani ed Harum, i principi figli del Califfo.

Le decisioni e i dialoghi con i personaggi sono controllate attraverso menù a scomparsa comandati dal mouse. Il joystick si rende invece indispensabile nei combattimenti, nella guida della nave e in altre situazioni di pericolo che via via si presentano durante il gioco.

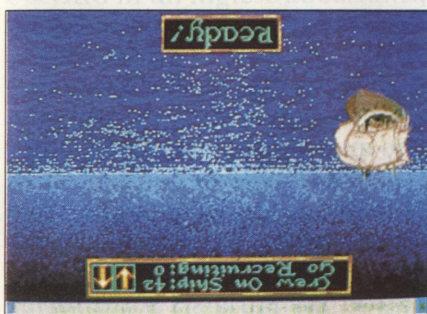
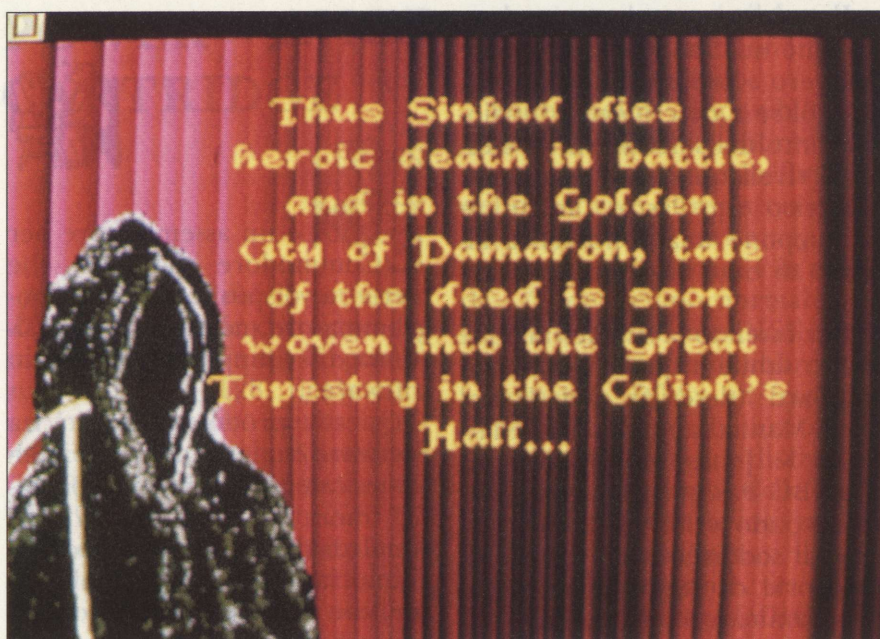
La schermata che appare all'inizio del gioco, raffigura una clessidra che segnala il tempo rimasto a disposizione prima che l'incantesimo che ha colpito il Califfo diventi irreversibile. Il tempo può essere visualizzato in qualsiasi momento tramite il menù «Show me». Quest'ultimo consente anche la visualizzazione di una mappa del mondo all'interno del quale Sinbad può viaggiare. Premendo il tasto sinistro del mouse si attiva una lente di ingrandimento che permette di leggere i nomi dei mari e delle località presenti sulla mappa.

Per i viaggi in mare Sinbad dispone di una nave (il Sabaralus) e di un equipaggio fidato. Ogni volta che si lascia un porto è possibile sguinzagliare una parte dei propri uomini alla ricerca di nuove reclute, mentre prima di attraccare bisogna decidere quanti uomini portare con se a terra e quanti lasciare a guardia della nave. La nave, infatti, può

essere attaccata dai pirati mentre si viaggia sulla terraferma, e l'eventuale perdita del natante può risultare catastrofica.

Un elemento che depone sicuramente a favore del programma risiede nell'estrema accuratezza con la quale è stata sviluppata la parte relativa ai duelli, che per il loro realismo ed agilità di controllo dei movimenti del contenedente non hanno nulla da invidiare a giochi dedicati espressamente al combattimento. È essenziale adottare una

buona tecnica nel duello. Le tattiche a disposizione sono quattro: colpo alto, colpo basso, avanzamento e difesa. I movimenti vengono controllati tramite joystick ed è necessario molto esercizio e prontezza di riflessi per risultare vittoriosi. Nel caso in cui Sinbad venga ucciso, il gioco termina ed appare una schermata con i risultati. Nel combattimento all'ultimo sangue col principe Nero o contro Jamaul non è possibile ritirarsi.

Non mancano spiacevoli incontri con pterodattili (spie del Principe Nero), che è necessario abbattere con una balestra, e con ciclopi che si possono uccidere con una fionda.

Durante i viaggi si possono incontrare diversi personaggi (elencati nel menù «Talk me»). I colloqui con ognuno di questi soggetti (che si conducono con l'aiuto del mouse) sono fondamentali ai fini della risoluzione del gioco. Vediamo qui di seguito di chi si tratta. LIBITINA: questa temibile seduttriche è conosciuta per i suoi poteri di magia nera. È una donna bellissima ed è l' ex-moglie del Califfo. Lasciandosi sedurre si ottiene in cambio potere. Libitina, inoltre, può indicare con precisione dove si trovino gli Idoli. Il PRINCIPE NERO: costui è, in realtà, uno dei figli del Califfo e per questo aspira a conquistare il trono. È fortissimo nel combattimento, può tentare di uccidere i principi Harum e Sylphani, e, se i suoi eserciti riescono a conquistare la capitale, diviene il nuovo Califfo, costringendo Sinbad ad un duello a morte con lui. IL GENIO: se lo si incontra (rinchiuso, naturalmente, in una lampada) sarà in grado di esaudire tre desideri. Lo SHAMAN: è l'unico che conosca il metodo per liberare il Califfo dall'incantesimo. GYPSY: si tratta di Iris, una vecchia zingara solitaria, che molti ritengono pazza. SYLPHANI e HARUM: i due principi seguono Sinbad in ogni luogo, ma, in generale, dai loro discorsi non può scaturire nulla di interessante. Gli IDOLI: si conosce molto poco riguardo alla leggenda degli idoli. Queste «statue», sono tre, sparse per il mondo, ogni statua ha due enormi brillanti al posto degli occhi. Si dice che colui che riuscirà a possedere un occhio di ognuno degli idoli conquisterà un enorme potere...

Dal menù «Show me» si può ottenere una mappa della città di Damaron e della relativa situazione strategica. Alla mappa è sovrapposta una griglia ad esagoni attraverso la quale si possono muovere le armate.

Sinbad è sicuramente uno tra i migliori giochi attualmente in commercio per Amiga. Esso dedica spazio a tutti i tradizionali campi dei «computer game», strategia, avventura ed azione, trasportando il giocatore in un'atmosfera fantastica, magica e coinvolgente, ed «obbligandolo», per la difficoltà di soluzione dei suoi misteri, a restare incollato al video per molte ore *(Amiga 1000, disco, $49.95).*

Mindscape Inc.
3444 Dundee Road
Northbrook, IL 60062
(001/312/4807677) USA

TURBO PASCAL

Ecco finalmente la possibilità di sfruttare il modo CP/M del 128 con un programma regolarmente in commercio!

Turbo Pascal è stato realizzato e prodotto dalla Borland International e viene distribuito in Italia dalla EDIA Bordland s.r.l. di Milano.

Il programma viene venduto insieme ad un ottimo manuale (un vero e proprio libro, n.d.r.), che, oltre a descrivere il funzionamento del software in se stesso, si rivela quasi uno strumento didattico sul Pascal, anche se presuppone una conoscenza almeno di base di questo linguaggio.

Pur essendo la sua definizione molto simile a quella del Pascal standard (descritta nel Pascal User Manual and Report redatto da K. Jensen e N. Wirth), il TURBO Pascal è stato progettato secondo criteri innovativi al fine di soddisfare le esigenze di categorie di utenti diverse, dal programmatore al neofita.

Questo linguaggio, concepito dal professor Niklaus Wirth, è sulla breccia da ormai parecchi anni, e la recente realizzazione di TURBO Pascal, oltre a permettere di sfruttare il tanto osannato (e trascurato) CP/M del 128, ne conferma ulteriormente la validità, l'affidabilità e la competitività *(C-128 in modo CP/M, disco + libro, Lit. 136.250).*

Edia Borland s.r.l.
V.le Cirene, 11
20135 Milano
(02/588523 - 5451953)

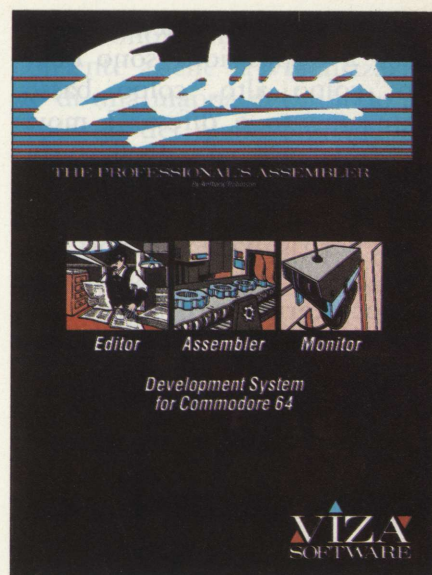
EDNA

EDNA è un assemblatore per Commodore 64/128 che mette a disposizione del programmatore medio un valido sistema di sviluppo per il proprio software.

Questo prodotto della Viza Software è uno tra i migliori della sua categoria ed offre possibilità finora riscontrabili solo su sistemi di dimensioni maggiori; a cominciare, ad esempio, dall'editor, che permette la creazione di un codice sorgente quasi con la stessa facilità di un word processor.

Tutti i comandi sono facilmente accessibili mediante la pressione di due tasti che, in generale, risultano di facile memorizzazione. L'assemblaggio procede ad una velocità notevole e incorpora numerose funzioni di utilità inseribili e disinseribili a piacere dell'operatore. È possibile scegliere se assemblare solo in memoria, solo su disco, se visualizzare la tabella dei simboli o se creare il noto listing dell'assemblaggio, che contiene informazioni riguardanti gli eventuali errori commessi e i codici esadecimali che si creano durante la compilazione. Tutte queste ultime possibilità possono, ovviamente, essere utilizzate anche in combinazione tra loro.

L'editor è stato realizzato con grande cura e fornisce comandi come la copia, la ricerca, lo spostamento, la sostituzione e la cancellazione di parole, di frasi o di blocchi. È dall'editor che si selezionano i dispositivi di I/O da utilizzare, si caricano, si salvano, si cancellano file e si accede ai comandi DOS. Sempre dall'editor si può concatenare tutto o parte del file in esame con altri file presenti su disco.

Le direttive di assemblaggio sono veramente numerose e versatili. I numeri possono essere espressi in notazione decimale,

esadecimale e binaria; una semplice funzione consente la conversione immediata da una base all'altra. L'area di editazione visualizza sulla parte alta dello schermo una barra di controllo che quantifica la memoria in uso ed è suddivisa nei consueti campi label, istruzione, operandi e commento. Le direttive consentono naturalmente di immettere dati in memoria sia sotto forma numerica che sotto forma di caratteri. Molte direttive sono curiose, perché mai riscontrate su altri assemblatori, come ad esempio l'assemblaggio condizionale, che consente di scegliere quale codice assemblare al verificarsi di una certa condizione. Questa possibilità può tornare utile per rendere semplice il passaggio su altri elaboratori, che si effettua semplicemente cambiando la condizione specificata; è poi il programma che sceglie automaticamente la parte di codice che si adegua al sistema in uso. A questo proposito il manuale per l'utente fa riferimento alla serie Commodore 8000, ma non si sofferma sul modo in cui sia possibile ottenere un file leggibile da tale sistema. Molto utili sono le direttive «.LIB,» e «.FIL,», che consentono di linkare librerie di simboli autocreate o di utilizzare quella presente sul disco fornito nel pacchetto, che contiene tutti gli indirizzi di entrata alle routine Kernal del 64

e altri indirizzi di varia utilità, come ad esempio i registri del colore e la locazione di inizio schermo.

Per lavorare a contatto diretto con la macchina è presente un monitor realizzato con grande cura, dotato di una funzione di breakpoint e di esecuzione a passi di un programma. È possibile in qualsiasi momento tornare al Basic riservando o, a richiesta, cancellando l'eventuale file sorgente presente in memoria. Una possibilità molto interessante offerta è quella della gestione di una scheda di programmazione per le memorie EPROM ed EAROM.

Il manuale, sebbene piuttosto conciso, è molto ben realizzato e scritto in un inglese abbastanza semplice. Esso descrive con chiarezza tutti i comandi disponibili e include numerosi consigli per il programmatore, descrivendo come utilizzare le routine create dal Basic e spiegando in modo abbastanza completo quali sono le aree di memoria utilizzate dalla cartridge. Per i possessori di Commodore 128 vi è una sezione dedicata alla descrizione dei comandi che il processore 65C02 mette a disposizione e che, naturalmente, sono disponibili anche in modo 64 e vengono riconosciuti dall'assemblatore.

Il disco fornito nella confezione comprende, oltre alla libreria Kernal 64, alcuni programmi di impostazione, orientati soprattutto alla gestione delle periferiche, e un programma di conversione che consente di utilizzare sorgenti create con Mikro Assembler.

Una nota di merito quindi alla Viza, che con questo nuovo prodotto mantiene alto il prestigio giustamente conquistato con le sue precedenti creazioni *(C-64/128 in modo 64, cartuccia + disco, Lit. 59.900).*

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

POWER CARTRIDGE

All'interno del vastissimo mercato delle cartridge destinate ad aumentare le potenzialità di C-64/128, finalmente un prodotto che merita realmente una particolare attenzione: Power Cartridge, realizzata nel 1986 dall'olandese KCS.

Questa cartuccia è dotata di un proprio sistema operativo di 16K, che non occupa nessuna parte della memoria del computer, e mette a disposizione i comandi Basic Toolkit, che semplificano notevolmente la programmazione, un caricatore veloce sia per disco che per nastro, un monitor per la programmazione in LM, un'interfaccia di stampa grafica molto ampia e la possibilità di copiare su supporto magnetico qualsiasi programma.

Vediamo ora, in rapida successione, le caratteristiche peculiari offerte dalla Power Cartridge. BASIC TOOLKIT: si tratta di un mezzo ausiliario per la programmazione in Basic, che mette a disposizione un buon numero di nuovi comandi direttamente all'accensione del computer, tra cui ricordiamo: Auto, per la generazione automatica dei numeri di riga; Color, semplifica il cambiamento dei colori di sfondo/testo; Deek (Doppio PEEK), consente di vedere contemporaneamente il contenuto di due indirizzi; Doke (doppio Byte Poke), per collocare un valore in due celle di memoria contemporaneamente; Find, per cercare un testo, un'istruzione o una variabile in un programma Basic; Hardcopy, per copiare il contenuto dello schermo su carta; e poi ancora MERGE, PAUSE, RENUM, REPEAT, QUIT, UNNEW...

Tutti i comandi sono semplici da utilizzare e di facile memorizzazione e comunque il loro uso è ben descritto dal buon manuale tradotto in italiano.

Sono inoltre disponibili una serie di comandi ed abbreviazioni atti a velocizzare le funzioni del disk drive di registrazione e caricamento. Il caricamento di un programma avviene in meno di 1/3 del tempo normale. Da notare è la funzione DISK (tasto F8), che permette di abbreviare i comandi DOS; ad esempio DISK "I" svolge la funzione di inizializzazione di un dischetto.

Questo prodotto della KCS contiene anche istruzioni per velocizzare il dialogo con il registratore. Interessante, fra gli altri, è il comando AUDIO, che rende visibile il segnale del registratore a cassette mediante linee tracciate sullo schermo video, trasmettendolo contemporaneamente all'altoparlante del televisore/monitor.

Non manca un discreto monitor/assemblatore (Powermon), richiamabile direttamente dal Basic o attraverso il tasto posto sul retro della cartuccia, con cui è possibile osservare, modificare e rilocare il contenuto di parti della memoria e, naturalmente, programmare in linguaggio macchina il 6510. Powermon non ha bisogno di nessuna area di memoria per funzionare e lascia a disposizione dell'utente tutta la memoria RAM della macchina. Vi sono inoltre molti comandi abbreviati che aumen-

tano notevolmente la facilità e la rapidità di utilizzo di questo assemblatore. Sul retro della cartridge si trova un interruttore di RESET. Premendo il pulsante si possono ottenere in qualsiasi momento le seguenti funzioni: CONTINUE, permette di continuare l'esecuzione di un programma precedentemente interrotto. BASIC, ritorna al Basic conservando tutte le variabili. RESET, è una normale funzione di reset. BACKUP DISK, è una funzione di Backup molto interessante che prevede diverse possibilità. RESET ALL, provoca il reset di qualsiasi programma, anche se dotato di una routine di autostart. BACKUP TAPE, svolge per i dati su nastro le stesse funzioni di backup disk. MONITOR, inserisce il Powermon. HARDCOPY, rende possibile l'output su stampante dello schermo video.

Power Cartridge, di cui abbiamo presentato solo una parte dei comandi disponibili, è in definitiva un valido prodotto, completo e ben curato nei suoi particolari. La confezione ed il manuale in italiano (stampato e non fotocopiato), leggibile e di facile comprensione, contribuiscono a conferire alla cartuccia olandese una posizione di rilievo nei confronti della concorrenza pirata *(C-64/128 in modo 64, cartuccia, Lit. 99.000).*

Lyndy-Elektronik
Via G. Ferrari, 14
22100 Como

SOFTWARE HELPLINE

Soluzione di OO-TOPOS

Prison Cell
Get bottle. Break lock (3 volte). Open door. West. North.

Guard Post
Push red. Push green. Get laser. Get goggles. East.

Hall Intersection
Shoot alien. North (3 volte). East. North. North. (On Podium) Get rod. South. South. West. South. South. South. West. South. South.

Medical Lab
Push switch. Get flask. Get box. Push switch. North. North. West. West.

Chemistry Lab
Push switch. Put flask in sink. Pull handle. Get flask. Push switch. East. East. South. West.

Small Room
Throw acid on sentinel. Shoot sentinel. Get translator. East. North. East. North. North. North. East.

Main Gravtube Room
Push button. Enter tube. Push blue. Exit. (Solarium) Get snarl. South. West. Down. Down.

Near Stage
Open box. Get converter. Get snarl. Wait (finchè Grix non vi butta fuori dalla stanza). Drop box. East. Enter tube. Push red (2 volte). Exit.

Garbage Disposal
Wear goggles. Push button. Search garbage. Get helmet. Wear helmet. Get suit. Wear suit. Get gloves. Wear gloves. Remove goggles. Exit. Push blue (2 volte). Exit. East.

Radiation Room
Put converter in beam. Get navchip. Get converter. East.

Bright Room
Wear goggles. Get sphere. Drop goggles. West (3 volte). South (3 volte). West. West. North.

Top of Tunnel
Drop navchip. Drop converter. Drop sphere. South. East. East. North. North. North. East. East. North. (Mirror Room) Look 4-D mirror. North.

Viewscreen
Push switch. Get card. Push blue. Push red (apre chest in frozen room). Push blue (3 volte). Push red (disattiva il tractor beam). Push switch. South. Look 4-D mirror. West. South. West. West. West.

Library
Push switch. Get crystal. Put crystal in projector. Push button. Get crystal. Open door. North.

Musty Room
Get book. Read book. Drop book. Push switch. South. East. East. South (3 volte). West. West. North.

Top of Tunnel
Drop crystal. Drop card. South. East. East. North. North. North. East. Enter Tube. Push blue. Exit. (Solarium) Open airlock. East.

Roof
Get ring. North. Yes. East. Up. (Top of Pyramid) Get jewel. Down. East. North.

Jungle Clearing
Get reed. North. (Dense Jungle) Touch flower. Get emerald. South. (Huja) Wave reed. Get gyro. East.

Beach (with Robot)
Throw reed in sea. Get shield. East.

Beach (with Crab)
(Se crab Vi morde, bevete liquid to heal.) Shoot crab. Drop laser. Get shell. Drop bottle. North. North. North.

Cargo Bay
Drop shell. Drop emerald. Drop gyro. Drop jewel. Drop ring. Drop translator. South. Open airlock. South. South. East. North. South.

Pyramid
Enter pyramid. Push switch. Get cube. Push switch. South. North. West. S.

Tunnel Bottom
Get in car. Push up button (2 volte). Get navchip. Get converter. Get crystal. Get card. Drop shield. Drop rod. Get sphere. Push down button (2 volte). North. East. East. West. North. North. North.

Cargo Bay
Drop navchip. Drop converter. Drop crystal. Drop card. Drop cube. Drop sphere. Get translator. South. South. South. East. North. West. South. Get in car. Push up (2 volte). Get Rod. Get shield. South. East. East. South.

Narrow Room
Push switch. Read writing. Taka ele-leva. Push switch. East. East.

Storage Room
Push switch. Get cylinder. Get purifier. Push switch. West. Down. Open door. South.
Cold Room
Push switch. Get vial. Push switch. North. Up. West. Taka ele-leva. Drop rod. Drop shield. Drop translator. North. West. West.
Lounge Area
Pull lever. Get block. North. Get in car. Push down (2 volte). North. East. East. West. North. North. North.
Cargo Bay
Drop cylinder. Drop purifier. Drop silver. Drop vial. South. South. South. East. North. West. South. Get in car. Push up (2 volte). South. East. East. South.
Narrow Room
Get rod. Get translator. Get shield. Taka ele-leva. East. Down. North.
Food Processing Room
Push switch. Get recirculator. Push Switch. South. Up. West. Taka ele-leva. North. West. West. North. Get in car. Push down (2 volte). North. East. East. West. North. North.
Inside Air Lock
Close airlock. North. (Cargo Bay) Drop translator. Drop suit. Drop helmet. Drop gloves. Get cylinder. South. West.
Computer Room
(Quando il computer richiede il mission code) TSE957X. North. (Life Support) Instail rod. Install recirculator. East.
Starboard Engine
Install shield. South. (Port Engine) Install cylinder. West. North.
Cargo Bay
Get purifier. Get converter. Get gyro. Get navchip. South. East.
Port Engine
Install purifier. Install converter. West. West. West. (Bridge) Install gyro. Install navchip. East. (Computer) Get message about needing 27,014 lemperes of fuel. East. North.
Cargo Bay
Get card. Get crystal. Get ring. Get shell. Get emerald. Get sphere. Get silver. Get jewel. Get cube. Get vial. South. West.
Install Card
(Get the message che dice che avete bisogno di 497.9 frods per comprare fuel e per sapere il prezzo di ogni articolo.) crystal - 0, ring - 31, shell - 133 emerald - 56, sphere - 89, silver - 69 jewel - 119, cube - 30. West.
Bridge
(Guardare il finale. Si vince se sia ha chiuso l'airlock, disattivato il tractor beam, si possiede vial e non sono stati richiamati i pirati.) ■

INPUT/OUTPUT SVELA I SEGRETI DEL VOSTRO COMPUTER

Input/Output rivela ogni mese ai suoi lettori routine di programmazione per C-64, C-128 ed Amiga. Avete in mente un'idea particolare, una routine, un utile stratagemma di programmazione, avete sviluppato un piccolo programma o in generale pensate che una Vostra nozione possa interessare l'utenza Commodore? Allora scriveteci, inviate il materiale a:

Commodore Gazette
Input/Output
Via Monte Napoleone, 9
20121 Milano

Sono grandemente apprezzati anche i contributi editoriali consistenti in articoli, foto, disegni... Scrivete a:

Commodore Gazette
Uffici Editoriali
Via Monte Napoleone, 9
20121 Milano

C-64/128 in modo 64

***093 Scomparsa di dati** - Nel C-64 è possibile nascondere istruzioni o parole riservate presenti su una linea, cioè far scomparire totalmente o parzialmente il contenuto di una linea di istruzioni Basic. Il procedimento da seguire è il seguente: si batta normalmente la riga, si metta poi in coda alla riga l'istruzione REM seguita dal simbolo «"» battuto due volte. Premere poi una volta il tasto INST DEL, digitando contemporaneamente SHIFT, e premerlo di nuovo tante volte quanti sono i caratteri contenuti nella riga (compresi l'apice e la REM). Si lasci poi il tasto SHIFT e si ripeta l'operazione, si prema cioè il tasto INST DEL tante volte quanti sono i caratteri della linea (appariranno delle T in reverse). A questo punto si prema RETURN e si richieda il listato: la linea in oggetto o risulterà sparita o si otterrà esclusivamente il numero di riga. Naturalmente questo procedimento non ha alcuna influenza sulle istruzioni della linea che al RUN verranno eseguite normalmente.

Antonio Di Palma
Nocera Inf. (SA)

***094 Macchina da scrivere** - Questa routine permette al C-64 di stampare una frase carattere per carattere, proprio come una macchina da scrivere. Il tutto avviene grazie alla funzione CHR$ della linea 20. Nella linea 30 sono racchiusi sotto forma di dati i codici ASCII di ciò che si vuole far stampare. Il ciclo FOR... NEXT della linea 20 introduce un ritardo tra una stampa e l'altra.

```
5 REM MACCHINA DA SCRIVERE
6 REM DI ANTONIO DI PALMA
10 READ A: IF A=-1 THEN END
20 PRINT CHR$(A)::FORT=1TO300:NEXT T:GOTO 10
30 DATA 147,87,32,67,79,77,77,79,68,79,82,69,
32,71,65,90,69,84,84,69 ,-1
```

Antonio Di Palma
Nocera Inf. (SA)

***095 Print AT** - La subroutine riportata qui di seguito consente la scrittura di un messaggio in qualsiasi parte dello schermo.

```
29999 END
30000 REM SUBROUTINE PRINT AT
30001 R9=VAL(MID$(M4$,1,2))
30002 C9=VAL(MID$(M4$,3,2))
30003 POKE 211,C9:POKE214,R9:SYS58640
30004 PRINT MID$(M4$,5,LEN(M4$))
30005 RETURN
```

Per poter scrivere un qualsiasi messaggio in una posizione stabilita dello schermo, ad esempio nella decima riga della diciassettesima colonna, è sufficiente inserire la seguente linea:

```
... M4$="1017 MESSAGGIO":GOSUB30000
```

Luigi Turchi
Spoleto

***096 Scritte verticali** - La routine qui di seguito riportata serve per far scrivere verticalmente ad un C-64 ciò che si digita. Non ha un utilizzo immediatamente pratico, ma può ad esempio essere inserita in un word processor per ottenere delle scritte speciali.

```
5 REM SCRITTURA VERTICALE
6 REM BY ANTONIO DI PALMA
10 GET A$:IF A$=""THEN10
20 IF ASC(A$)=13 THEN END
25 PRINT A$
30 GOTO 10
40 REM CON RETURN SI ESCE
```

Antonio Di Palma
Nocera Inf. (SA)

***097 Routine grafiche** - Questi programmi consentono rispettivamente la creazione di uno sprite tramite tastiera e la produzione di un'onda in movimento.

```
15 REM ********************
16 REM * GIOVANNI CATTINA  *
20 REM *  VIA TEMPIO N 9   *
21 REM * 07014 OZIERI (S.S)*
22 REM ********************
25 PRINT"■■TASTO F1 CAMBIA COLORE SPRITE"
26 PRINT"■TASTO F3 ESPANSIONE SPRITE IN X"
27 PRINT"■TASTO F4 FINE ESPANSIONE IN X"
28 PRINT"■TASTO F5 ESPANSIONE SPRITE IN Y"
29 PRINT"■TASTO F6 FINE ESPANSIONE IN Y"
30 PRINT"■TASTI CURSORE MOVIMENTO SPRITE"
100 V=53248:X=150:Y=157:POKEV+32,14:POKEV+33,14:
   POKE650,128
105 FORI=16000TO16061:READA:POKEI,A:NEXTI
1000 POKE2040,250:POKEV+21,1
1005 POKEV,X:POKEV+1,Y:C=C+1:IFC=255THENC=0
1010 GETA$:IFA$="▶■"THENX=X+1
1020 IFA$="■"THENX=X-1
1030 IFA$="■"THENY=Y+1
1040 IFA$="■"THENY=Y-1
1050 IFA$="■"THENPOKEV+39,C
1055 IFA$="■"THENPOKEV+29,7
1056 IFA$="■"THENPOKEV+29,0
1060 IFA$="■"THENPOKEV+23,7
1065 IFA$="■"THENPOKEV+23,0
1085 IFX=-1THENX=0
1086 IFX=256THENX=255
1090 IFY=-1THENY=0
1095 IFY=256THENY=255
1100 GOTO1005
2000 DATA 0,0,0,0,120,0,0,132,0,0,204,0,0,132,0,
   0,180,0,0,120,0,0
2010 DATA48,0,1,254,0,1,74,0,1,74,0,1,74,0,3,75,
   0,0,120,0,0,48
2020 DATA0,0,72,0,0,132,0,0,132,0,1,134,0,0,0,0,
   0,0,0
```

```
0 REM ******************************
1 REM *  PROGRAMMA PER C.B.M 64/128 *
5 REM *         B.G.IANNESSA        *
6 REM *       VIA MANNU N. 12       *
7 REM *    07023 CALANGIANUS (S.S)  *
8 REM *       NUM.TEL.079/661173    *
9 REM ******************************
10 :
20 GOSUB200
30 CE=53280:CI=CE+1:CC=646:S=1024:G=1
40 POKECE,0:POKECI,0:POKECC,4:S=1024
50 FORA=0TO69STEP.199
60 B=SIN(A)
65 IFG=2THENIFB=.197689165THENG=1
70 PRINTTAB(10+E+B*10)"■■■"
80 C=C+1:POKECC,C:IFC=15THENC=1
90 GOSUB120
100 NEXTA
110 GOTO50
120 IFG=1THEND=D+1:IFD=HTHEND=0:E=E+1
130 IFG=2THEND=D+1:IFD=HTHEND=0:E=E-1
140 IFE=18THENGOSUB160
150 RETURN
160 G=2:F=F+1:IFF=20THENG=1:F=0
170 RETURN
200 INPUT"DURATA DELL'ONDA";H
210 H=H+10
220 PRINT"■";
230 RETURN
300 REM ##################################
```

Giovanni Cattina
Ozieri (SS)
Benito Iannessa
Calangianus (SS)

Commodore 128

***098 Labirinth** - Questo programma è utile per comprendere alcuni dei principali trucchi applicabili alla creazione di sprite ed animazioni. Per giocare utilizzate i tasti 1, 3, 7 e 9.

```
10 GRAPHIC 1,1:DRAW 1,12,8:SSHAPE SP$,0,0,21,24:
   SPRSAV SP$,2
20 COLOR 0,1:COLOR 4,1:COLOR 5,8:COLOR 1,4
30 SPRITE 2,0
40 GRAPHIC0,1:PRINT"L A B I R I N T H !"
50 PRINT"■BY CILLO & GEJITI"
60 PRINT"■ISTRUZIONI :"
70 PRINT"SEI STATO TRASFORMATO IN UN ATOMO"
80 PRINT"E DEVI USCIRE DAL LABIRINTO SENZA"
90 PRINT"TOCCARE I MURI. PENA ARROSTIMENTO"
100 PRINT"PERCHE' NEI MURI CI PASSANO 20"
110 PRINT"MILA (?) VOLTS. HAI 3 ATOMI E"
120 PRINT"23 LIVELLI DA SUPERARE."
130 PRINT"PREMI UN TASTO PER INIZIARE."
140 GETKEYQW$
150 LE=1:BN=1500:OM=0
160 FAST
170 COLOR 1,14:X=159+13:Y=LE*8+60
180 GRAPHIC1,1:B=1
190 FOR A=0 TO 39
200 C=INT(RND(1)*2+1)
210 IF C=1 THEN CHAR 1,A,B,"\"
220 IF C=2 THEN CHAR 1,A,B,"/"
230 NEXT
240 B=B+1
250 IF B<=LE THEN 190
260 SPRITE 2,1
270 SLOW
280 GETA$
290 IF A$="1"THEN X=X-1:Y=Y+1
300 IF A$="3"THEN X=X+1:Y=Y+1
310 IF A$="7"THEN X=X-1:Y=Y-1
320 IF A$="9"THEN X=X+1:Y=Y-1
330 IFX>317+13THENX=317+13
340 IFX<15 THEN X=15
350 IFY<44 THEN 550
```

```
360 IFY>196+42 THEN Y=196+42
370 MOVSPR 2,X,Y
380 COLLISION 2,400
390 GOTO 280
400 J=J+1:IFJ=2THENBEGIN:J=0:VOL2
410 FOR A=23000 TO 29000 STEP 1500
420 SOUND 1,A,1
430 NEXTA
440 OM=OM+1:IF OM=3 THEN BEGIN
450 OM=0:GRAPHICO,1
460 PRINT"I TUOI ATOMI SONO TUTTI"
470 PRINT"MORTI. IL TUO PUNTEGGIO FINALE"
480 PRINT"E' DI";PB;"PUNTI":SLEEP6:RUN 30
490 BEND
500 BEND
510 X=159+13:Y=LE*8+60
520 IF A$="9"THEN X=X-1:Y=Y+1
530 MOVSPR 2,X,Y
540 RETURN
550 SLEEP 1:VOL 1:SPRITE2,0: GRAPHICO,1
560 PRINT"HAI COMPLETATO IL LABIRINTO !!"
570 SLEEP 1
580 BN=BN+100:PB=PB+1500+BN
590 FOR E=0 TO BN STEP 100
600 SOUND 1,2000,6
610 PRINT"BONUS : ";E
620 NEXTE
630 PRINT"PUNTI :";PB
640 IF LE=23 THEN BEGIN
650 PRINT"HAI RISOLTO TUTTI I LIVELLI !!"PRINT"
    PUNTI TOTALI =";PB
660 SLEEP 5:RUN 30
670 BEND
680 LE=LE+1:PRINT"PASSI AL LIVELLO";LE
690 SLEEP2:GOTO 160
```

Francesco Chiello
Milano

Commodore Amiga

***099 Rappresentazione grafica di una funzione** - Lo studio di una funzione matematica è probabilmente uno dei campi più interessanti per l'utilizzo del calcolatore. La possibilità, data una funzione, di poterne visualizzare il grafico è estremamente utile in quanto è possibile trarre informazioni immediate studiando il suo andamento.

Il programma scritto in AmigaBASIC qui riportato produce la rappresentazione grafica in alta risoluzione dell'inviluppo di una funzione. La prima riga di programma pulisce lo schermo e predispone una zona sufficientemente ampia per permettere l'utilizzo dell'alta risoluzione. A questo proposito si noti che nella riga successiva l'istruzione SCREEN presenta, come valori per la dimensione della finestra, 640 per la lunghezza e 500 per l'altezza; questo è un fatto molto interessante in quanto benché il manuale fornito in dotazione presenti come valori massimi per uno schermo in alta risoluzione 640x400 pixel è possibile, ma solo con la versione 1.2 del Kickstart, aprire una finestra di 640x500 pixel, usufruendo così della totalità dello schermo. Le linee 30-40 convertono le misure degli angoli in radianti mentre le righe 50-76 selezionano i colori per lo sfondo, per la rappresentazione degli assi e dei punti della funzione. Nella riga 100 viene definita la funzione, la quale può essere modificata a piacere. In proposito è necessaria una precisazione: così come è definito, il programma rappresenta solo funzioni simili a quella data. Se si vogliono visualizzare altri tipi sono necessarie alcune modifiche atte ad evitare messaggi di «overflow». I cambiamenti riguardano sostanzialmente le coordinate del centro (110-120), i fattori di scala (130-140) e gli incrementi necessari per il tracciamento del grafico. È inoltre possibile variare tipo di risoluzione semplicemente modificando i parametri del comando SCREEN e le linee 81-82. Se per esempio si vuole una risoluzione di 320x200 sono necessarie le seguenti modifiche:

```
10 SCREEN 1,320,200,4,1
81 LUNGMIN = 0 : LUNGMAX = 319
82 ALTMIN = 0 : ALTMAX = 199
```

Anche con questo tipo di risoluzione è possibile incrementare l'altezza della finestra sino a 250 pixel mantenendo il controllo del gadget per il suo spostamento e modifica.

Per finire, un consiglio per chi volesse ottenere un output su stampante. Nella versione 1.2 del Workbench è presente, nella directory SYSTEM, una utility per stampare le schermate. Per utilizzarla al meglio è sufficiente spostare l'icona di GRAPHICDUMP in alto a destra e poi rimpicciolire la finestra di SYSTEM. Una volta fatto ciò è sufficiente caricare l'AmigaBASIC e il programma, dando come valori per la finestra in cui verrà visualizzato il grafico, quelli ammessi sul manuale (640x400, 320x200, ecc). In questo modo sarà possibile visualizzare la finestra di SYSTEM, contenente GRAPHICDUMP, nelle ultime righe di schermo. Per stampare il grafico sarà sufficiente, una volta terminata l'esecuzione del programma, accendere la stampante e «cliccare» il pulsante del mouse sopra l'icona relativa all'utility per stampare la schermata.

```
5 CLS:CLEAR 30000  : REM alloca memoria per il programma
10 SCREEN 1,640,500,2,4   :REM parametri per
11 WINDOW 4,"funzioni",,7,1 :REM l'alta risoluzione
20 DEFINT h-n
30 pi = 4 * ATN (1)
40 dr=pi/180
50 REM definizione dei colori
60 PALETTE 2,.93,.2,0
61 PALETTE 0,.13,.57,.6
75 PALETTE 1,.33,.87,0!
76 PALETTE 3,0!,0!,0!
80 REM margini della finestra
81 lungmin=0 :  lungmax=639
82 altmin=0 :   altmax=499
83 hres=lungmax-lungmin : lres=altmax-altmin
90 REM definizione della funzione e relativi parametri
100 DEF FNf1(x,Y)=SQR(x*x+Y*Y-3)
105 REM coordinate del centro
110 xcentro=lungmin+INT(hres/2)
120 ycentro=altmin+INT(lres/2)
125 REM dimensionamento della funzione
130 scalah=3
140 scalav=3
160 c1=ATN ((ycentro-altmin)/(lungmax-xcentro))
170 c2=pi-ATN ((ycentro-altmin)/(xcentro-lungmin))
180 c3=pi+ATN ((altmax-ycentro)/(xcentro-lungmin))
190 c4=2*pi-ATN ((altmax-ycentro)/(lungmax-xcentro))
210 COLOR 2,3
```

```
220 INPUT "angoli per asse x,y,z",gradx,grady,gradz
230 CLS
235 REM routine per il tracciamento degli assi cartesiani
240 gradi=gradx
250 GOSUB 1080
260 sinxa=SIN(a) : cosxa=COS(a)
270 GOSUB 1010
280 gradi=grady
290 GOSUB 1080
300 sinya=SIN(a) : cosya=COS(a)
310 GOSUB 1010
320 gradi=gradz
330 GOSUB 1080
340 sinza=SIN(a) : cosza=COS(a)
350 GOSUB 1010
360 REM routine di tracciamento della funzione
370 FOR ty=-20 TO 20 STEP 2
420 FOR tx=-20 TO 20 STEP .3
421 IF tx*tx+1>ty*ty THEN 550 ELSE tz=FNf1(tx,ty)

440 GOSUB 1230
441 IF tx<0 THEN colore=1 ELSE colore=2
450 PSET (mh,mv),colore
460 tz=-tz:GOSUB 1230
540 PSET (mh,mv),colore
550 NEXT tx,ty
560 END
1000 REM
1010 tana=TAN(a)
1020 GOSUB 1170 :h1=hz : l1=lz
1030 gradi=gradi+180 : GOSUB 1080:
1040 GOSUB 1170 : h2=hz : l2=lz
1060 RETURN
1080 WHILE gradi<0
1090 gradi=gradi+360
1100 WEND
1110 WHILE gradi>=360
1120 gradi=gradi-360
1130 WEND
1140 a=gradi*dr
1150 RETURN
1170 IF a<=c1 OR a>c4 THEN hz=lungmax:lz=ycentro-(lungmax-
     xcentro)*tana:RETURN
1180 IF a<=c2 THEN lz=altmin : hz=xcentro+(ycentro-altmin)
     /tana:RETURN
1190 IF a<=c3 THEN hz=lungmin :lz=ycentro+(xcentro-lungmin)
     *tana: RETURN
1200 IF a<=c4 THEN lz=altmax :hz=xcentro-(altmax-ycentro)
     /tana
1210 RETURN
1230 px=tx*cosxa+ty*cosya+tz*cosza
1240 py=tx*sinxa+ty*sinya+tz*sinza
1250 mh=INT(xcentro+px*scalah)
1260 mv=INT(ycentro-py*scalav)
1270 RETURN
```

MODIFICHE DA APPORTARE AL PROGRAMMA PER OTTENERE IL GRAFICO DI UN PARABOLOIDE.

```
100 def FNf1(x,y)=(5*x*x+2*y*y-4)
110 xcentro=altmin+int(hres/2)
120 ycentro=altmin+int(19*lres/20)
130 scalah=12
140 scalav=2
370 for ty=-6 to 6 step .4
420 for tx=0 to 6 step .4
430 tz=FNf1(tx,ty)
440 gosub 1230
450 pset (mh,mv),2
460 next tx
510 for tx=-6 to 0 step .15
520 tz=FNf1(tx,ty)
530 gosub 1230
540 pset(mh,mv),3
550 next tx,ty
560 end
1250 mh=int(xcentro+px*scalah)
1260 mv=int(ycentro-py*scalav)
```

Giorgio Papetti
Milano

***100 Stampanti -** Riportiamo qui di seguito un elenco di stampanti compatibili con il Commodore Amiga.

Stampanti a colori:

Stampante	Driver	Commenti
Canon PJ1080A	Canon PJ1080A	Richiede il custom driver 3
Diablo C150	Diablo C150	
Epson JX-80	Epson JX80	
Apple Image Writer II	Image Writer II	Richiede il custom driver 2
Juki 5510	Epson JX-80	Richiede l'upgrade colore
NEC CP2 & CP3	Epson JX-80	
Okimate 20	Okimate 20	Richiede l'Amiga plug «N»

Stampanti a matrice di punti:

Stampante	Driver	Commenti
Alphacom Alphapro	Alphacom Alphapro 101	
Apple Image Writer	Image Writer	Richiede il custom driver 3
Brother 1509	Epson	
Citizen MSP Series	Epson	
C. Itoh 8510	Pro Driver	Richiede il custom driver 5
CIE CI-3500	CBM MPS1000	Rich. il modulo IBM comp.
Serie Epson FX	Epson	
Serie Epson LX	Epson	
Serie Epson MX	Epson	
Serie Epson RX	Epson	
IBM Graphics (5152)	CBM MPS1000	
Juki 5510	Epson JX-80	
Okidata 182 & 183	CBM MPS1000	In modo IBM
Okidata 192 & 193	CBM MPS1000	In modo IBM
Okidata 292 & 293	CBM MPS1000	Rich. IBM Plug «N»
Panasonic 1080	Epson	
Panasonic 1091	Epson	
Panasonic 1092	Epson	
Panasonic 1592	Epson	
Radio Shack DMP 300	CBM MPS1000	
Serie Star SG	CBM MPS1000	Non sono supportate tutte le caratteristiche
Serie Star SD & SR	CBM MPS1000	Non sono supportate tutte le caratteristiche
Serie Star MX	Epson	In modo Epson comp.
Serie Star NB	CBM MPS1000	In modo IBM comp.
Serie Star NL	Epson	Rich. il modulo Espon comp.

Stampanti a margherita:

Stampante	Driver	Commenti
Brother HR-15XL	Brother HR-15XL	
Citizen Premier 35	Diablo 630	
Comrex CR-II	Brother HR-15XL	
Comrex CR-III	Brother HR-15XL	
Commodore 6400	Diablo 630	Richiede l'opt. Centronics
Diablo Advantage D25	Diablo Advantage D25	
Diablo 630	Diablo 630	
Dynax	Brother HR-15XL	
Epson DX-20	Diablo 630	
Juki 6000	Diablo 630	
Juki 6100	Diablo 630	
Juki 6300	Diablo 630	
Qume LetterPro 20	Qume LetterPro 20	
Panasonic KXP 3131	Diablo 630	
Panasonic KXP 3151	Diablo 630	
Tec F10	Diablo 630	

Stampanti laser:

Stampante	Driver	Commenti
Canon Laser	Epson	In modo FX-80
CIE Lips 10	Epson	In modo Epson
	Diablo 630	In modo Diablo 630
H.P. Laser Jet	H.P. Laser Jet	
H.P. Laser Jet Plus	H.P. Laser Jet Plus	

Note:

1 - Tutte le stampanti elencate richiedono un cavo appropriato.
2 - La Redmond Cable ha realizzato un driver Apple Image Writer (tel. 001/206/8682168).
3 - Contattare la Tycom Technologies (tel. 001/216/2617088).
4 - La V.1.2 del Workbench include il driver Oki 292.
5 - $20.95 presso: Micro-Cybernetics, PO BOX 3126, Laurel, MD 20708, USA. ■

Il Consumer Electronics Show

Servizio speciale sulla nota rassegna svoltasi a Las Vegas (8-11 gennaio).

Di Matthew Leeds

L'Amiga collegato all'espansione PAL della Byte By Byte e ad una tastiera all'interno dello stand della Commodore

I quattro giorni nei quali si è articolata l'ultima edizione del CES (Consumer Electronics Show, n.d.r.), tenutasi a Las Vegas in Nevada nel mese di gennaio, sono stati testimoni della presentazione di numerose ed interessanti novità.

Per i non addetti ai lavori è forse necessario ricordare che il Consumer Electronics Show è la più ampia ed importante manifestazione di prodotti elettronici tra tutte quelle che si svolgono negli Stati Uniti. In essa vengono presentate tutte le novità audio, video, televisive e, naturalmente, anche tutti quei prodotti legati a questo tipo di industria. Ogni anno la **Commodore** sottolinea la sua presenza con un grande stand nel quale introduce al pubblico le sue più recenti novità, invitando nel contempo

a partecipare a questa manifestazione anche un buon numero di aziende produttrici di software ed hardware dedicato.

Le novità di quest'anno sono state numerose e tutte di grande importanza tecnica. È stato innanzitutto presentato ufficialmente sul mercato americano il **PC10**. Questa macchina è già da qualche tempo commercializzata sul mercato europeo e su quello canadese con buoni risultati di vendita. Negli USA sarà venduta una versione dell'elaboratore dotata di porta parallela e seriale, scheda grafica mono/colore (compatibile con IBM, CGA, MDA, Hercules e Planatronics), floppy disk drive, CPU 8088 a 4.77 MHz, MS-DOS 3.2, BIOS XT compatibile, tastiera modello XT, GW BASIC e di cinque slot di espansione. Saranno inoltre disponibili due versioni del medesimo computer: il PC10-1, dotato di un solo floppy e di 512K di RAM, ed il PC10-2, che possiede invece due floppy e 640K di memoria RAM. Per entrambi i calcolatori in questione è inoltre prevista l'istallazione di hard disk.

Un altro elaboratore Commodore la cui vendita era finora riservata al mercato europeo è l'ormai noto 128D, la cui introduzione negli Stati Uniti è stata pianificata per il mese di aprile. Le altre novità includono una versione definitiva del disk drive da 3.5" **1581**, le cui capacità di immagazzinamento dati sono di 800K, ed una nuova espansione di memoria per il 64C di 256K RAM, la 1764, il cui problema di alimentazione è stato risolto dalla realizzazione di un nuovo trasformatore. Il package include anche due programmi: un disk driver RAM per il sistema operativo originale ed uno per le applicazioni di GEOS.

Dalla **Berkeley Softworks** continuano a giungere buone notizie sul favore di utenti trovato da GEOS. A dimostrazione di ciò possiamo portare anche le notevoli dimensioni dello spazio espositivo di questa azienda nell'ambito del CES, in occasione del quale sono state presentate versioni funzionanti di **geoCalc** e **geoBase**, insieme ad una versione di GEOS ad 80 colonne per il C-128. Sembra infatti che la Berkeley e la Commodore si siano accordate per risolvere, con successo, il problema del display video per questa macchina. Non è ancora stata fissata una data per la commercializzazione del prodotto, pur rimanendo come assai probabile quella di fine maggio.

La **Electronic Arts** ha presentato **Earl Weaver Baseball** per Amiga. Si tratta di un ottimo gioco, dotato di display a schermi separati, guidato da tattiche «intelligenti», con possibilità di ottenere replay istantanei e lanci personalizzati. Non mancano anche suoni digitalizzati uniti a molte altre caratteristiche interessanti. Oltre a questo programma è da segnalare una ver-

L'area espositiva dell'Aegis Development, il produttore di alcuni tra i migliori pacchetti grafici per Amiga

sione non ancora definitiva del nuovo **Deluxe Video,** dalla quale è stato eliminato il bordo «overscan» ed è stata implementata nel contempo un'animazione molto più precisa. Ha avuto già inizio, infine, la commercializzazione di Deluxe Music.

Kickstart Eliminator costituisce il pacchetto sul quale la **Creative Microsystems** punta maggiormente. Si tratta di una serie di Eprom, PAL e jumper, che, una volta installati sull'Amiga, permettono di non dover più inserire il disco del Kickstart all'accensione del computer. Rimane comunque necessario, nel caso il programma lo richieda, l'inserimento del Workbench. Utilizzando il software incluso nel package, è possibile servirsi della RAM WCS come memoria, aggiungendo altri 256K a quella «fast» già disponibile. È indispensabile che l'utente sia in grado di istallare da solo i componenti di questo package ed è necessaria una certa abilità nel saldare e nel maneggiare chip.

La **Prism Computer Graphics** sta portando a termine la realizzazione di un **«SMPTE time code reader»** per Amiga, il quale consente di leggere un codice SMPTE direttamente dalla fonte, e viene commercializzato unitamente al software che permette di creare delle «decision list off-line». È possibile selezionare i punti interni ed esterni visivamente. È consentito anche il trasferimento via RS232 dell'EDL su altri computer o CMX. Il pacchetto permette di visualizzare i punti di inizio e di fine degli edit da nastro, di controllare la continuità del time code e di visualizzarlo sullo schermo.

Fra i prodotti per C-64/128 la **Simon&Schuster** ha presentato la versione per il 1987 del programma di J.K. Lasser: **Your Income Tax**. Il software è strutturato in modo tale da seguire passo per passo l'utente nel processo inerente la decisione su quali matrici immagazzinare e quali informazioni inserire, permettendo di realizzare contemporaneamente diverse proiezioni sui possibili risultati delle operazioni. Sono presenti schermi di help ed un calcolatore, mentre la possibilità di aprire finestre rende molto semplice la gestione del programma.

L'ormai noto in Italia C-128D rappresenta una novità per gli USA

La **Microcube Corp.** ha presentato una serie di «add-on» fra i più interessanti tra quelli sviluppati negli ultimi tempi. Essa, infatti, commercializza dei joystick, disegnati sulla base della strumentazione di un vero aeromobile, utili per giocare con Flight Simulator. La versione **ATC** del joystick è dedicata a C-64/128 e permette il controllo diretto dei flap, dei freni, del motore, delle mitragliere, del timone, degli alettoni e del trim. La versione per Amiga fornisce in più i comandi per il radar, per le finestre relative alla visione dell'aereo, per il carrello di atterraggio, per le bombe, per l'accensione del motore, per la radio, per il transponder, per le luci e per alcune altre funzioni. Il joystick può essere regolato in modo da tornare automaticamente in posizione neutra e può essere utilizzato come un mouse, nel caso in cui lo spazio per usare quello originale si riveli insufficiente. Vengono anche offerte delle possibilità per i programmatori che intendono sviluppare applicazioni per questa periferica.

La **Microsoft Press** ha presentato il suo nuovo volume **«L' AMIGA: Immagini, suoni e animazioni sul Commodore Amiga»**. La bellissima copertina dell'opera merita sicuramente di essere menzionata insieme all'ottima qualità dei contenuti. La spiegazione relativa al modo in cui il computer gestisce grafica e suono è in assoluto la più chiara che abbia mai avuto modo di leggere. All'interno si trovano anche esempi di programmazione in AmigaBASIC utili per gestire BOB, sprite, suoni, finestre ed animazione. Se si desidera conoscere più dettagliatamente il funzionamento interno della macchina e/o si è interessati alla programmazione in AmigaBASIC, questo libro si rivela davvero indispensabile. La versione italiana di questo volume sarà pubblicata dalla **IHT Technologies** (l'editore di Commodore Gazette, n.d.r.) e sarà disponibile a partire dal mese di maggio.

La **Infinity Software** era presente con **Grand Slam Tennis** per Amiga. Il fatto che uno dei programmatori che hanno realizzato il gioco sia un mio amico non mi impedisce di elencarne in modo obiettivo le principali peculiarità, relative ad uno sfruttamento adeguato del mouse, una buona animazione, un' ottima digitalizzazione dei suoni, un sufficiente numero di livelli per mantenere un buon grado di

difficoltà ed una curata veste estetica della confezione.

Oltre a quelli sin qui citati, sono stati introdotti al pubblico diversi altri prodotti degni di interesse. Tim Jenison della **NewTek** ha presentato delle versioni non ancora definitive del software di sua produzione dedicato a **DigiView**; la **Anakin Research** era presente al CES con **Easyl**; Karen Bayless ha dimostrato le capacità di **ProWrite**; la **Mimetics** ha realizzato uno «studio di registrazione» quasi completo; la **Liquid Light** ha esposto un riproduttore di film; la **Byte by Byte** disponeva del suo nuovo orologio/calendario, **TIC**, per Amiga ed infine la Commodore era presente anche con una versione del **Sidecar** con approvazione **FCC** (la necessità dell'approvazione del prodotto contro interferenze radio è stata la causa del grande ritardo nella sua uscita sul mercato USA, n.d.r.). Si sono nuovamente ammirate anche le demo riguardanti **Genlock e Live!**, un frame grabber dotato di nuove capacità software.

In definitiva un bilancio più che positivo per un mondo di tecnologie che non finisce mai di accattivarsi il favore del pubblico con novità sempre più interessanti.

Per ulteriori informazioni contattare direttamente:

***Creative Microsystem,* Inc**
10110 SW Nimbus Street
Tigard, OR 97223
(001/503/6203821) USA

Prism Computer Graphics
14755 Ventura Blvd
Suite 1-600
Sherman Daks, CA 91403
(001/818/9069977) USA

Simon&Schuster
One Gulf + Western Plaza
New York, NY 10023
(001/212/3738882) USA

MicroCube Corp.
POB 488
Leesburg, VA 22075
USA

Infinity Software
1331 61st Street
Emeryville, CA 94608
(001/415/4201551) USA ■

Nuovi nati nella famiglia Commodore: *Amiga* 500 e 2000

Servizio speciale in esclusiva sui nuovi elaboratori della famiglia Amiga presentati in anteprima mondiale alla stampa europea alla fine di gennaio nell'incantevole scenario delle isole Canarie.

Fuerteventura, isola dell'arcipelago delle Canarie, è stata la cornice nella quale, al villaggio Club Aldiana presso Playa de Jandia, ha avuto luogo la presentazione alla stampa europea degli ultimi nati della linea Amiga: il 500 ed il 2000. La manifestazione, articolata in tre giorni di seminari e workshop, svoltisi tra il 27 ed il 29 gennaio, è stata organizzata dalla Commodore Büromaschinen di Francoforte, in Germania, presso la quale hanno sede gli uffici dirigenziali della CBM per tutta l'Europa e gli unici laboratori di ricerca al di fuori degli Stati Uniti, nei quali hanno preso vita prodotti come il PC10/20/40, il Sidecar e lo stesso Amiga 2000. Sono in-

Il villaggio Club Aldiana di Fuerteventura sede della manifestazione

tervenuti alla presentazione trentasei giornalisti provenienti da numerosi Stati d'Europa, ma il nucleo principale era costituito da una fitta rappresentanza della stampa tedesca. L'unica rivista italiana invitata è stata, a parte una dedicata al mondo dei PC in generale, Commodore Gazette.

Amiga 500

Quello che le prime indiscrezioni, sulla base della passione bellica di uno dei suoi progettisti, avevano annunciato come B-52, dal nome del noto bombardiere, si è poi rivelato sotto il nome di Amiga 500 essere una macchina con 512K di RAM del tutto compatibile con la serie

1000. Si potrebbe parlare di un Amiga 1000 con tastiera, unità centrale e disk drive da 3.5" integrati in un' unica struttura il cui design ricorda molto quello di un C-128 sovradimensionato.

In alto: il nuovo Commodore Amiga 500. Notare il disk drive laterale e la tastiera espansa. In basso: l'Amiga 2000. Il suo corpo macchina richiama da vicino quello del PC40

L'hardware della macchina è rimasto il medesimo con la differenza che la continua evoluzione della tecnologia ha permesso un'elevata integrazione dei circuiti le cui dimensioni minori hanno consentito la realizzazione di un'unità dalle dimensioni ridotte. La tastiera non più separabile dal corpo centrale è stata resa più pratica e contiene 94 tasti, cinque in più dell'Amiga 1000. Il disk drive interno è sempre da 3.5" ed è locato sul lato destro dell'elaboratore nel quale non è più presente lo slot di espansione (presente invece su quello sinistro, n.d.r.). Due led, posti a poca distanza dalla tastierina numerica, segnalano rispettivamente l'alimentazione e l'operatività del drive.

L'A500, come l'A1000, dispone di un'interfaccia parallela, una seriale e dei connettori per disk drive esterni e per il supporto delle uscite audio e video.

I 512K di meoria RAM possono essere espansi internamente fino ad 1MB. Esternamente la macchina può essere espansa fino a 9MB. La politica di contenimento delle dimensioni ha imposto l'ausilio di un trasformatore esterno.

La caratteristica più eccezionale dell'A500 è senza dubbio il suo prezzo, che è stato annunciato essere sotto i 500 dollari e che, per una macchina sostanzialmente identica all'A1000, è veramente incredibile. Non esiste attualmente sul mercato nessun computer in grado di reggere al rapporto prezzo/prestazioni dell'A500 che con il costo annunciato si colloca nella fascia di prezzo delle macchine ad 8 bit.

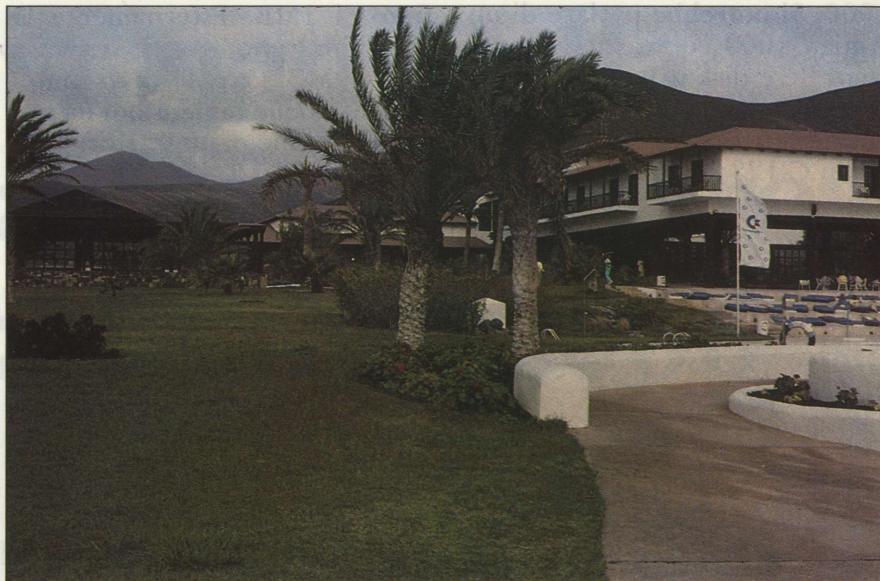
Se fino ad oggi le vendite dell'Amiga non potevano essere confrontate ai milioni di unità C-64 vendute nel mondo, il nuovo prezzo dell'A500 gli apre senza dubbio la strada a diventare il «best seller» della fine degli anni '80 e a detenere con un livello qualitativo e tecnologico decisamente superiore lo scettro avidamente impugnato per tanto tempo dal C-64.

Per quel che riguarda la disponibilità di questo elaboratore si è parlato di una prima apparizione sul mercato per il mese di maggio, ma dal momento che l'A500 è un progetto interamente di West Chester (la sede dei laboratori Commodore statunitensi, n.d.r.), e quindi made in USA, e che i prodotti provenienti dagli Stati Uniti arrivano in Europa sempre con un certo

ritardo rispetto a quelli sviluppati in Germania (cfr. C-128D, Sidecar, PC10/20/40 ed anche Amiga 2000, n.d.r.), riteniamo che l'A500 verrà proposto in Italia, a Milano, in occasione dello

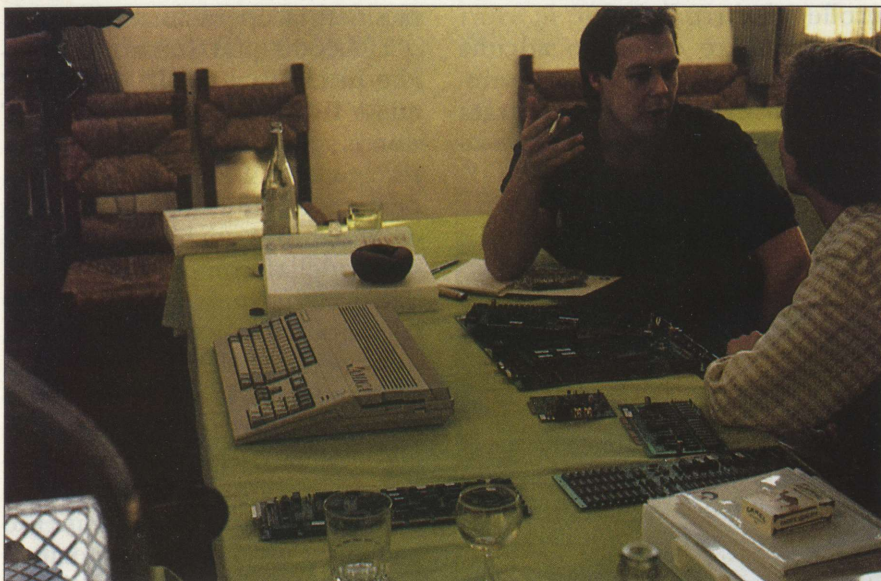
SMAU, nel mese di settembre.

Amiga 2000

Sulla base del successo suscitato dall'introduzione sul mercato dell'Amiga si sono venute delineando due esigenze distinte, rispecchianti rispettivamente il desiderio di prezzi più accessibili e la necessità di funzioni per un uso professionale. La prima richiesta verrà ampiamente (forse anche oltre le aspettative più ottimistiche) soddisfatta dall'A500, mentre alla seconda vengono incontro le peculiarità di una nuova macchina, nata per combinare tra loro le eccezionali capacità dell'A1000 con la flessibilità di un sistema con un'architettura aperta simile al PC10/20/40, un computer completamente compatibile con l'A1000, la cui configurazione può essere adattata per incontrare le richieste particolari di qualsiasi utente: l'Amiga 2000 (e non 2500 come era stata inizialmente chiamata questa macchina, n.d.r.).

In alto: il Club Aldiana presso Playa de Jandia. In basso: una delle sale nelle quali hanno avuto luogo le conferenze ed i workshop

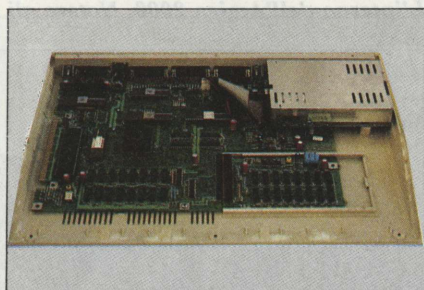
Questo modello incorpora non solo le prestazioni della CPU 68000 e quelle audio/video già proprie dell'A1000, ma anche un esclusivo sistema di schede addizionali, che consentono di configurare la macchina secondo modalità operative diverse, come ad esempio la compatibilità IBM PC XT ed AT, ed un orologio calendario con batteria (che evita l'impostazione data/ora, n.d.r.).

Nella sua versione standard l'A2000 dispone di 1MB di memoria RAM, espandibile internamente ad 8.5MB. Il Kickstart V1.2. è ora contenuto in 256K ROM e non necessita più così di essere caricato all'accensione dell'elaboratore.

La tastiera è stata ampliata e dispone di 96 tasti, sette in più dell'A1000, per lo più preposti alla compatibilità IBM, grazie ai quali non sono più necessarie le combinazioni di tasti richieste dal Sidecar, per ottenere funzioni come Prt Sc, Num Lock o

L'Amiga 500, la scheda madre e diversi slot di espansione per A2000

Scroll Lock. Un'altra peculiarità non indifferente della nuova tastiera risiede nella sua stessa struttura, basata su quella delle tastiere di PC10/20/40 che hanno una risposta allo sfioramento dei tasti tra le migliori in assoluto tra tutte quelle dei PC presenti sul mercato (molto superiore alla «classica» risposta dei tasti dei computer IBM, n.d.r.), che consente ore di uso piacevole, pratico e non affaticante per l'operatore.

Oltre al disk drive interno standard da 3.5" si può istallare un altro drive interno da 3.5", o un hard disk drive interno da 3.5", e un disk drive interno da 5.25" o hard disk drive interno da 5.25". Esternamente si possono aggiungere altri due drive da 3.5" o da 5.25", oltre a quelli direttamente controllati attraverso la porta di espansione della scheda per la compatibilità IBM.

La struttura del calcolatore è nel suo complesso formata da una solida unità centrale in metallo di dimensioni superiori a quella dell'A1000, che ricorda da vicino quella del PC40 o dello stesso IBM AT, e da una tastiera esterna non riponibile sotto il corpo macchina (come avviene per quella dell'A1000, n.d.r.). All'interno del corpo centrale trovano posto, come abbiamo già detto, tre disk drive o hard disk dive, ed un massimo di nove schede. Sette dei relativi slot sono destinati ad espansioni di memoria, controller di periferiche e coprocessori, uno è disegnato per l'inserimento di un'al-

In alto e in basso: l'interno dell'A500

tra CPU, come ad esempio il 68020/68881, e l'ultimo è destinato ad applicazioni video, concretizzabili nell'istallazione del modulatore video che sull'A2000 è opzionale. Due led, locati nella parte frontale sinistra dell'unità, segnalano rispettivamente l'alimentazione ed il funzionamento dell'hard disk. Il trasformatore interno è stato potenziato rispetto a quello dell'A1000 e dispone ora di 200W, necessari del resto ad alimentare i numerosi disk drive e le varie schede.

Non manca la consueta abbondanza di porte di comunicazione con il mondo esterno. Oltre alle due porte per mouse/joystick e ad il connettore della tastiera, locati nella parte frontale della macchina, sono presenti, posteriormente, le porte video RGB, parallela Centronics IBM compatibile, disk drive, audio stereo e seriale IBM compatibile. Nell'A2000 le porte esterne sono state adattate agli standard correnti (ed infatti sono diventate IBM compatibili, n.d.r.) e non necessitano più di connettori speciali. Oltre alle porte qui citate dobbiamo ricordare che anche ogni singola scheda può interfacciarsi con l'esterno attraverso propri connettori per i quali è previsto l'input/output attraverso appositi sportellini. È inoltre necessario specificare che l'introduzione degli slot di espansione interni ha causato l'eliminazione dello slot esterno collocato nell'Amiga 1000 sul lato destro del corpo macchina.

Il prezzo dell'A2000 dovrebbe essere inferiore ai 1500-2000 dollari e la sua introduzione sul mercato italiano è prevista per il mese di maggio.

La particolare struttura espandibile propria di questo elaboratore, legata alla compati-

Un A2000 corredato con due disk drive da 3,5" ed un hard disk da 5.25"

bilità IBM PC XT ed AT la cui richiesta da parte di una certa fascia di utenti era grandissima, e lo dimostrano gli ampi sforzi progettuali della Commodore sfociati in prodotti più o meno felici come Sidecar e Transformer, e l'architettura interna della macchina, meritano un discorso a parte.

Architettura interna, slot ed A2088

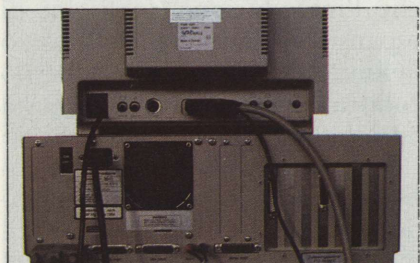
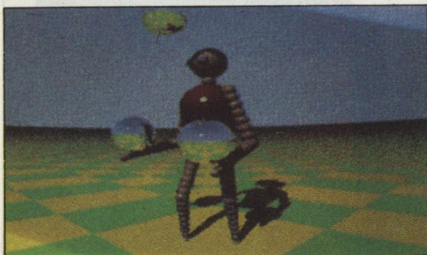
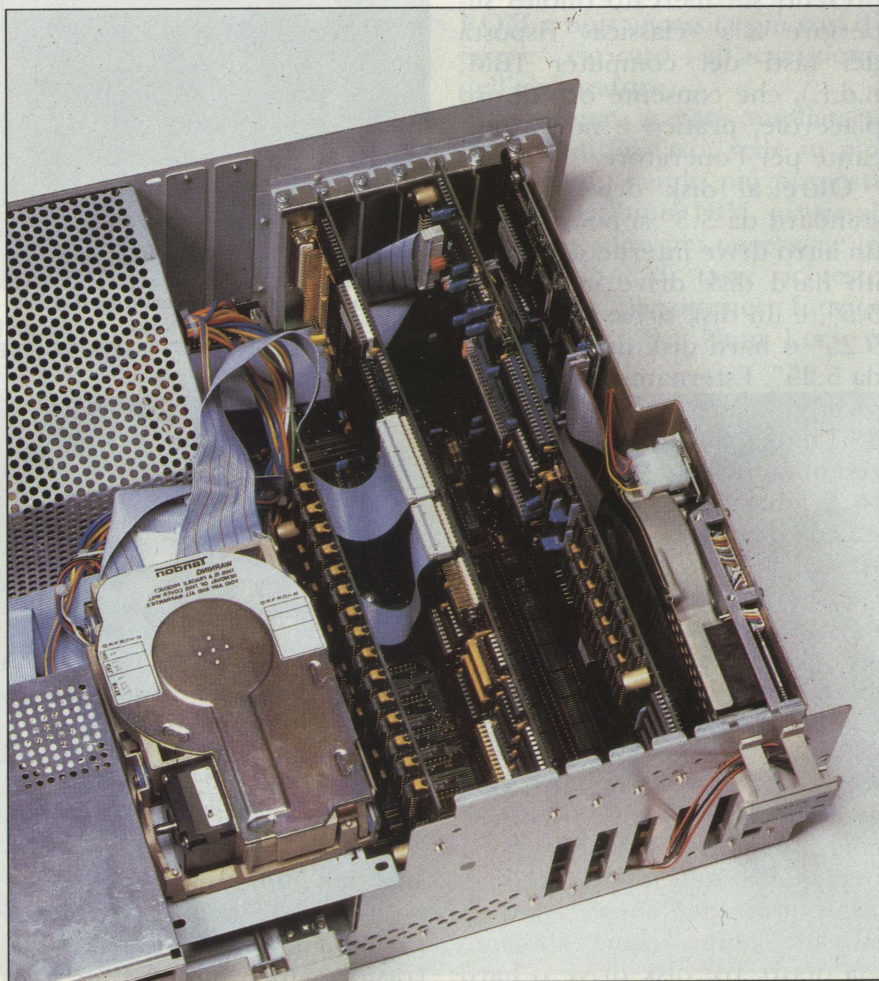
Il sistema minimo A2000 (il cui prezzo dovrebbe essere inferiore ai 1500 dollari, n.d.r.) è comparabile ad un A1000 con 1MB di RAM ed un orologio/calendario in tempo reale. Gli slot permettono di adattare e di espandere la macchina secondo le proprie esigenze. Questo può concretizzarsi nell'inserimento di espansioni di memoria e nell'istallazione di coprocessori e controller per hard disk (come l'A2094 hard disk/SCSI controller, n.d.r.).

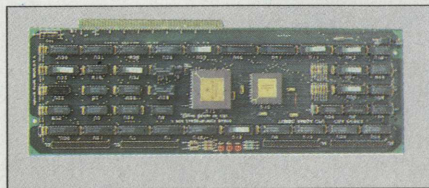
Uno slot è destinato ad una CPU addizionale, come ad esempio la 68020/68881, realizzata dalla Computer System Associates.

I disk drive da 3.5" dell'A2000

È importante a questo punto far notare che una eventuale seconda CPU non esclude il 68000 originario con il quale può lavorare raggiungendo una completa sinergia operazionale.

L'A2000 può funzionare anche in emulazione IBM PC XT ed AT. Mentre la scheda AT non è stata ancora messa a punto, quella XT, la A2088, è già perfettamente funzionante. Una delle caratteristiche più interessanti della compatibilità IBM dell'A2000 consiste nel fatto che il computer non si trasforma in un comune IBM compatibile, perdendo così la sua reale identità. Al contrario, l'utente non si accorge neanche del funzionamento di un programma in formato MS-DOS, dal momento che quest'ultimo appare normalmente in una finestra del Workbench. Lo stesso sistema operativo ed il 68000 dell'Amiga continuano a funzionare in contemporanea con il microprocessore Intel 8088 con cui dialogano attraverso un vero e proprio ponte tra i due sistemi (la scheda è infatti collocata in parte in uno

La parte posteriore dell'Amiga 2000

In alto: un'applicazione di videodigitalizzazione in tempo reale su A2000. In basso: una demo grafica delle potenzialità dell'accoppiata MPU compressori

L'interno dell'Amiga 2000. Notare gli slot e la schermatura del trasformatore

slot di espansione a 100 pin Amiga ed in parte in uno slot PC AT, n.d.r.), chiamato Ianus (dal lat. ianus = apertura che mette in comunicazione due luoghi attraverso un passaggio coperto, n.d.r.), dal nome dell'antica divinità romana la cui statua bifronte era collocata sulla Porta Ianualis, con le due facce rivolte rispettivamente ad oriente ed occidente. Sulla base di questo si può addirittura affermare che il funzionamento in MS-DOS del 8088 non è che una delle applicazioni possibili, dal momento che l'8088 può, attraverso Ianus, essere usato per «fare dei favori alla CPU 68000», come ha detto l'ingegnere della Commodore tedesca Torsten Burgdorf nel corso della presentazione a Fuerteventura, lavorando in contemporanea all'AmigaDOS nella gestione di un programma.

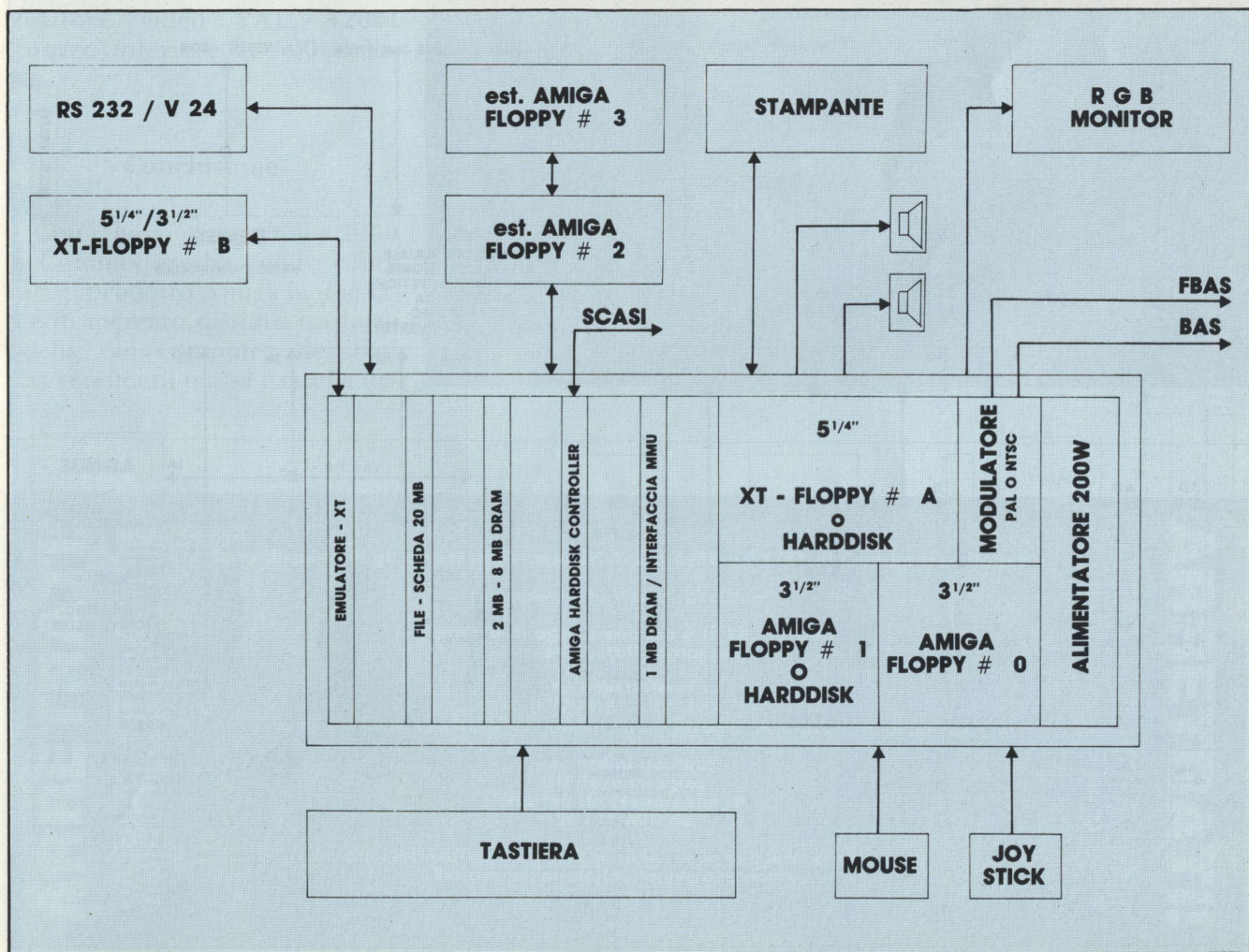
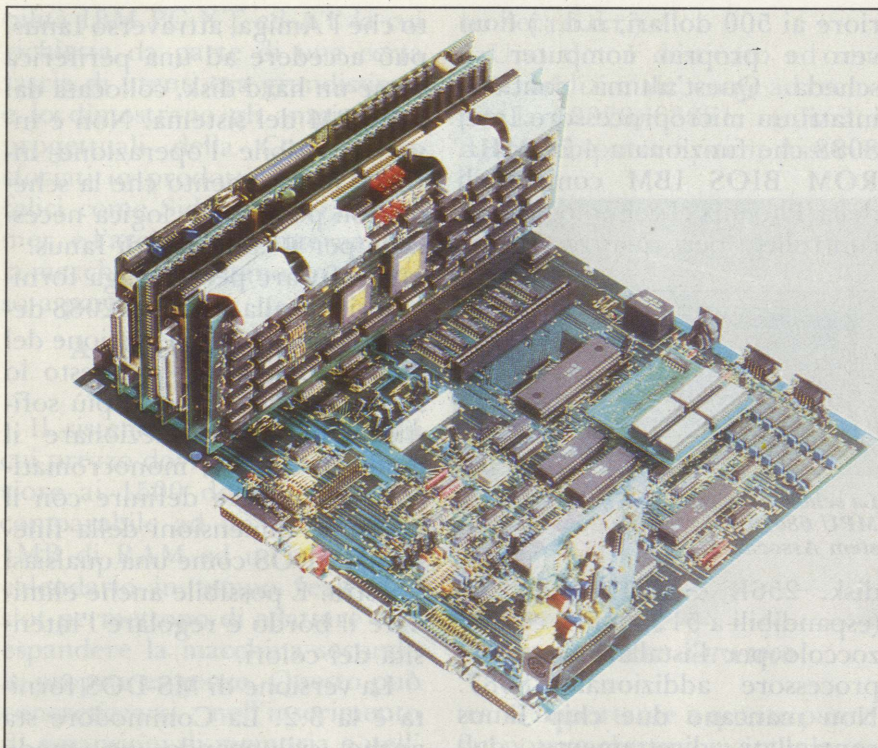
Il board A2088 (prezzo inferiore ai 500 dollari, n.d.r.) è un vero e proprio computer su scheda. Quest'ultima contiene infatti un microprocessore Intel 8088 che funziona a 4.77 MHz, ROM BIOS IBM compatibili della Phoenix Technologies, un controller per quattro floppy

La scheda di espansione per A2000 con MPU 68020 68881 della Computer System Associates

disk, 256K di RAM standard (espandibili a 512, n.d.r.) ed uno zoccolo per l'istallazione di un processore addizionale 8087. Non mancano due chip Ianus controllati direttamente dall'Amiga.

Una caratteristica interessante di questo sistema risiede nel fatto chė l'Amiga, attraverso Ianus, può accedere ad una periferica come un hard disk, collocata dal lato IBM del sistema. Non è invece possibile l'operazione inversa, dal momento che la scheda non possiede la logica necessaria per il controllo di Ianus.

Il software per l'Amiga fornito insieme alla scheda A2088 deriva da quello per la gestione del Sidecar, ma, come del resto lo stesso sistema, è molto più sofisticato. Si può selezionare il modo di display, monocromatico o a colori, e definire con il mouse le dimensioni della finestra MS-DOS come una qualsiasi finestra. È possibile anche eliminare il bordo e regolare l'intensità dei colori.

La versione di MS-DOS fornita è la 3.2. La Commodore sta anche sviluppando un modo operativo per lo scambio in tempo reale di file tra MS-DOS ed AmigaDOS.

RS 232 / V 24
est. AMIGA FLOPPY # 3
STAMPANTE
R G B MONITOR
5 1/4"/3 1/2" XT-FLOPPY # B
est. AMIGA FLOPPY # 2
SCASI
FBAS
BAS
EMULATORE - XT
FILE - SCHEDA 20 MB
2 MB - 8 MB DRAM
AMIGA HARDDISK CONTROLLER
1 MB DRAM / INTERFACCIA MMU
5 1/4" XT - FLOPPY # A O HARDDISK
MODULATORE PAL O NTSC
ALIMENTATORE 200W
3 1/2" AMIGA FLOPPY # 1 O HARDDISK
3 1/2" AMIGA FLOPPY # 0
TASTIERA
MOUSE
JOY STICK

La scheda madre dell'A2000 con schede di espansione e modulatore PAL

Nuovi prodotti per la linea Amiga

La Commodore ha finalmente dedicato alla linea Amiga due nuove stampanti la **MPS 2000** e la **MPS 2000C** (una stampante a colori) per le quali però non è stata specificata la data di uscita sul mercato.

Il problema della vibrazione del display nel modo 640 x 400 interlaced sarà risolto dall'introduzione del nuovo monitor (RGB digitale, RGB analogico, composito, n.d.r.) **A2080** ad alta persistenza il cui prezzo sarà inferiore ai 500 dollari.

Tra gli altri prodotti annunciati ricordiamo: monitor a colori **A2002** (RGB analogico, RGB digitale, composito), disk drive esterno da 3.5" **A1010**, disk drive esterno da 5.25" **A1020**, scheda di espansione per compatibilità IBM PC XT **A2088** (prezzo

STAMPANTE
FLOPPY est. int.
RS 232 AUDIO MOUSE JOY STICK
VIDEO - RGB
MOD. VIDEO
Connettore - 62 pin PC
PC
Connettore - 62 pin PC
Conn. 36 pin
AT
Conn. 36 pin
BATTERIA
OROLOGIO IN TEMPO REALE
PORTA PARALELLA
PORTA FLOPPY
PAULA
INTERFACCIA SERIALE
INTERFACCIA MOUSE
INTERFACCIA JOYSTICK
INTERFACCIA AUDIO STEREO
CON 4 D/A CONVERTER
DENISE
VIDEO CONTROLLER
D D D
IA ID IA ID
Connettore - 100 pin AMIGA
Connettore - 100 pin AMIGA
Connettore - 86 pin MMU
AD AD D D D
DATA BUFFER
AC AC C C C
CONTROL BUFFER
AA AA A A A
ADDRESS BUFFER
CPU 68000
D
DATA BUFFER
ID
IA<1:8>
ID<0:15>
IA
ID
AGNUS
BIT DMA CONTROLLER
BLITTER
GRAPHIC CONTROLLER
IA
ADDRESS BUFFER
A
ADDRESS MUX
KICK START ROM
D
A A
ADDRESS MUX
ID
DRAM
CHIP - RAM
512K * 8 BIT
A<1:23>

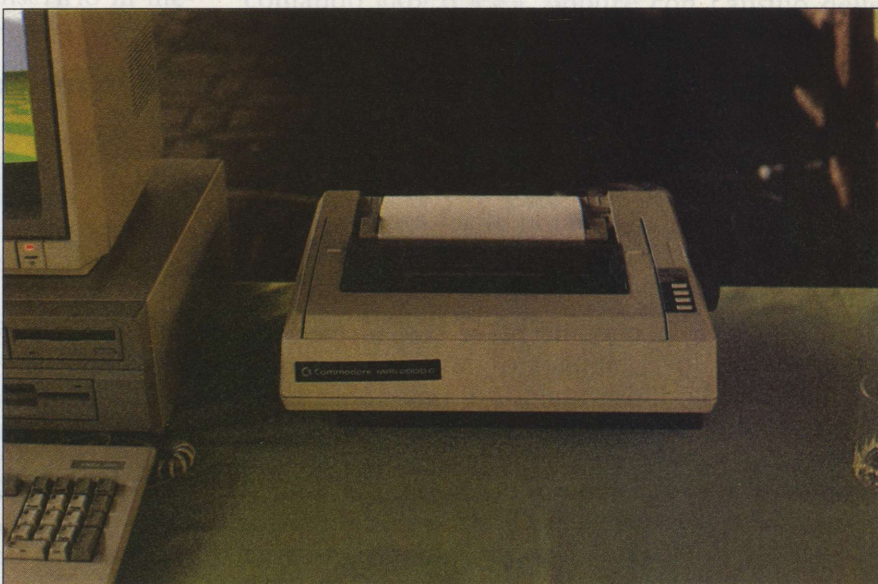
In alto: la nuova stampante MPS 2000. In basso la stampante a colori MPS 2000C

inferiore ai 500 dollari), hard disk/SCSI controller **A2094** (controlla due drive ST506 e sette device SCSI), espansione di memoria 2MB **A2050**, espansione di memoria 8MB **A2058**, modulatore video PAL **A2061** (prezzo inferiore ai 500 dollari).

Conclusione

Con i nuovi Amiga 500 e 2000 la Commodore ha voluto collocare il prodotto Amiga in due fasce di mercato distinte tra loro: quella dei computer destinati alla vendita di massa e quella degli elaboratori professionali.

L'A500 è senza dubbio il prodotto che ha tutte le carte in regola per eguagliare, se non superare, lo strepitoso successo suscitato dal C-64, con un rapporto qualità/prezzo che non ha sicuramente precedenti nella breve storia dei personal computer.

L'A2000 non è solo un Amiga 1000 potenziato, è bensì una macchina che risponde in pieno a quelle che dagli ingegneri della Commodore sono state declamate come le caratteristiche del computer del futuro, multitasking, multiprocessor e multi-DOS. La nostra precisazione può essere quella che l'A2000 non è una tecnologia del futuro: è una realtà di oggi! ■

Nuove istruzioni del chip 8502 utilizzato come CPU nel Commodore 128

Un'eccezionale ed esclusiva raccolta di istruzioni inedite per l'8502.

Di Marco Menichelli

Alcuni mesi fa ero alle prese con un vecchio programma in Linguaggio Macchina. Come ognuno ben sa, l'inserimento di una nuova linea contenente delle istruzioni non risulta essere in L.M. un' operazione altrettanto semplice quanto sarebbe se il programma fosse in Basic. L'Editor del L.M. in dotazione al C-128 non contempla infatti tale possibilità. Occorre quindi ovviare a questo inconveniente con il comando Transfer, trasferire appunto una certa parte del programma, aggiungere le istruzioni necessarie per poi ritrasferire ed allacciare nuovamente le istruzioni precedentemente spostate in altra zona di memoria. Non è un' operazione difficile ad eseguirsi: occorre solo prestare un po' di attenzione e ricontrollare il programma prima di attivarlo. Se si ha fretta di renderlo operativo, si può tornare al Basic impartendo poi la fatidica SYS di partenza.

Il programma in oggetto, al contrario di quanto si potesse pensare, pur funzionando perfettamente, non sortiva gli effetti desiderati. Mi trovavo senz'altro di fronte ad un errore di concetto; pertanto mi accinsi a rientrare nel Monitor ed a disassemblare la modifica effettuata. Dopo alcune linee apparvero sullo schermo dei punti interrogativi, che il disassemblatore è solito mostrare beffardamente all'operatore nel momento in cui rileva un operando non preceduto da alcuna istruzione. L'inconveniente si era presentato durante l'operazione di Transfer e la porzione del programma trasferito, avendo fornito all'Editor un indirizzo errato, non si era integrata correttamente con il resto del programma. Pertanto ciò che era un operando, si era, senza dubbio, trasformato in un'istruzione... e così per ben due volte.

Giunsi alla conclusione che l'8502 è in grado di riconoscere altri due codici esadecimali, oltre a quelli già noti a tutti i disassemblatori ed assemblatori relativi alla famiglia del 6502: il primo è un NOP che salta due celle di memoria, oltre alla propria, che si può chiamare «NP3», e l'altro, che conferisce all'accumulatore A ed al registro X lo stesso valore dell'operando che lo segue, si può chiamare «AXL».

Stanti queste premesse decisi che, in relazione al programma che stavo esaminando, era opportuno scandagliare tutti i numeri compresi fra 0 e 255, scartando quelli che già costituivano un'istruzione nota. Inserii la presunta istruzione in $0B00, facendola seguire da una serie di zero (BRK), modificai con il comando «R», l'accumulatore ed i registri, ed impartii successivamente il comando G F0B00. Ottenni sempre dei risultati interessanti e quando il programma non si bloc-

cava ero in grado di verificare cosa era accaduto attraverso la lettura del valore dei registri. Attraverso il Program Counter ero anche in grado di rilevare il numero dei byte interessati dalla nuova istruzione.

In seguito, con un po' di pazienza, riuscii a decifrare e a codificare una quarantina di nuove istruzioni che, utilizzate con i molteplici loro indirizzamenti, costituiscono un insieme di 89 (sono documentate nella Tavola presente alla fine di questo articolo, n.d.r.). Ciascuna di esse è denominata con una sigla di tre lettere di semplice assimilazione, come avviene solitamente anche per le istruzioni standard del 6502.

Alcune di esse sono davvero molto interessanti e, se usate frequentemente in un programma, permettono di ottenere un sensibile risparmio di memoria. Infatti, a seconda dell'istruzione e del suo utilizzo, si può arrivare ad un risparmio da 1 a 6 byte per ogni nuovo codice.

Un esempio è dato dal seguente programma, peraltro banale, che stampa sullo schermo 256 volte il carattere «?». Provate ad inserirlo in memoria utilizzando il comando «>»:

```
>F0B00 A9 3F 85 FA 20 D2 FF C7 FA D0 F9 60
```

Provando a disassemblarlo ci si trova di fronte ad un «listato» che, data la presenza di un paio di punti interrogativi, sembra non aver significato. Il programma, a dispetto delle apparenze, opera invece correttamente e lo si può constatare impartendo in diretta i seguenti comandi: BANK 15 : SYS DEC ("B00") e quindi RETURN.

In questo modo abbiamo imparato ad utilizzare l'istruzione C7, che corrisponde a DEM $aa. Per ottenere lo stesso risultato con i «vecchi» codici, si sarebbe dovuto redigere un programma di questo tipo:

```
>F0B00 A9 3F 85 FA 20 D2 FF C6 FA C5 FA D0 F7 60
```

occupando, evidentemente, due byte di memoria in più.

Anche NP2 ed NP3, le istruzioni più banali, possono essere molto utili se impiegate nel modo seguente:

```
................................
0C05    BEQ     $0C0E
0C07    BVC     $0C11
0C09    BCS     $0C14
0C0B    LDY     #$01
0C0D    NP3
0C0E    LDY     #$02
0C10    NP3
0C11    LDY     #$03
0C13    NP3
0C14    LDY     #$04
0C16    ..................
................................
```

Se le istruzioni precedenti alla linea 0C05 riportano in Status il valore zero settato, si salta alla 0C0E dove Y assume il valore di 02 rimanendo tale fino alla linea 0C16, in quanto le istruzioni NP3 delle linee 0C10 e 0C13 fanno saltare le linee 0C11 e 0C14. Lo stesso succede anche per le altre condizioni di salto. È altrettanto vero che l'istruzione BIT potrebbe essere utilizzata in sostituzione di NP3, ma solo nei casi in cui non si sia interessati all'alterazione del registro di stato.

Si rivela a questo punto superfluo citare altri esempi, dal momento che grazie alla Tavola dei nuovi codici riportata qui di seguito dovreste essere in grado di riconoscere voi stessi il loro possibile impiego nell'ambito delle singole cessità specifiche.

Naturalmente, dal momento che l'assemblatore del C-128 non sa riconoscere il significato di tali istruzioni, non è in grado nè di leggerle nè di scriverle; per superare questo ostacolo è sufficiente inserirle in memoria con il comando «M» o «>», oppure con l'ausilio delle semplici POKE.

Nella Tavola, nella colonna «Corrisponde a:», è stato riportato un esempio che si riferisce solo al modo di indirizzamento in pagina zero. Naturalmente, sono validi tutti gli indirizzamenti riportati nella colonna «Mnemonico». A questo proposito sarà facile notare che tali nuovi codici propongono alcuni tipi di indirizzamento (indiretto indicizzato, indicizzato indiretto e assoluto indicizzato in Y) non previsti per le normali istruzioni come ASL, ROL, LSR...

Particolare interesse è da concedere alle istruzioni SAX e ASB in virtù della loro potenza.

È risultato inoltre assai difficile decodificare le seguenti istruzioni:

02, 12, 22, 32, 42, 52, 62, 72, 82, 92, 9B, B2, C2, D2, F2

Perché non provate voi?

Tavola dei nuovi codici del chip 8502

NP1 = NOP

Identico al NOP. N. 1 BYTE. Passa all'istruzione successiva: 1A , 5A , 7A , DA , FA .

NP2

Anche questa istruzione non esegue alcuna operazione, ma salta una cella di memoria in più della precedente. N. 2 BYTE:
04 , 14 , 34 , 44 , 54 , 64 , 74 80 , 89 , 93 , D4 , E2 , F4.

NP3

Analoga alle precedenti, ma salta due celle di memoria. N. 3 BYTE:
0C , 1C , 3C , 5C , 7C , DC , FC .

SOR = ASL + OR

Il contenuto della Memoria in indirizzo subisce un ASL. L'acc. A riporta il risultato dell'OR fra Memoria e acc. stesso:

Codice	Mnemonico	N° BYTE	Corrisponde a:
03	SOR ($aa,X)	2	ASL $aa
07	SOR $aa	2	ORA $aa
0F	SOR $aaaa	3	
13	SOR ($aa),Y	2	
17	SOR $aa,X	2	
1B	SOR $aaaa,Y	3	
1F	SOR $aaaa,X	3	

DPM = ROL + AND

Il contenuto della Memoria in indirizzo subisce un ROL (avviene cioè un raddoppio tenendo conto del carry). A riporta il risultato dell'AND fra Memoria ed acc. stesso:

Codice	Mnemonico	N° BYTE	Corrisponde a:
23	DPM ($aa,X)	2	ROL $aa
27	DPM $aa	2	AND $aa
2F	DPM $aaaa	3	
33	DPM ($aa),Y	2	
37	DPM $aa,X	2	
3B	DPM $aaaa,Y	3	
3F	DPM $aaaa,X	3	

LEO = LSR + EOR

Il contenuto della Memoria in indirizzo subisce un LSR. L'acc. A riporta il risultato dell'EOR fra la Memoria e l'acc. stesso:

Codice	Mnemonico	N° BYTE	Corrisponde a:
43	LEO ($aa,X)	2	LSR $aa
47	LEO $aa	2	EOR $aa
4F	LEO $aaaa	3	
53	LEO ($aa),Y	2	
57	LEO $aa,X	2	
5B	LEO $aaaa,Y	3	
5F	LEO $aaaa,X	3	

ROA = ROR + ADC

Il contenuto della Memoria in indirizzo subisce un ROR. L'acc. A riporta il risultato della somma fra il contenuto della Memoria e l'acc. stesso:

Codice	Mnemonico	N° BYTE	Corrisponde a:
63	ROA ($aa,X)	2	ROR $aa
67	ROA $aa	2	ADC $aa
6F	ROA $aaaa	3	
73	ROA ($aa),Y	2	
77	ROA $aa,X	2	
7B	ROA $aaaa,Y	3	
7F	ROA $aaaa,X	3	

SAX = Store A AND X

Immette nella Memoria in indirizzo il risultato dell'AND fra A ed il registro X:

Codice	Mnemonico	N° BYTE	Corrisponde a:
83	SAX ($aa,X)	2	STX $aa
87	SAX $aa	2	PHA
8F	SAX $aaaa	3	AND $aa
97	SAX $aa,Y	2	STA $aa
9F	SAX $aaaa,Y	3	PLA

AXL = A & X Load

Immette il contenuto della memoria in indirizzo sia in A quanto nel registro X:

Codice	Mnemonico	N° BYTE	Corrisponde a:
A3	AXL ($aa,X)	2	LDA $aa
A7	AXL $aa	2	TAX
AB	AXL #	2	oppure
AF	AXL $aaaa	3	LDA # $xx
B3	AXL ($aa),Y	2	TAX
B7	AXL $aa,Y	2	
BF	AXL $aaaa,Y	3	

L'istruzione AB è incostante. Talvolta si perde il BIT 4 di A e X.

AXP = Load A, X & SP

Immette il contenuto della Memoria in indirizzo in A, nel registro X e nello Stack Pointer:

Codice	Mnemonico	N° BYTE	Corrisponde a:
BB	AXP $aaaa,Y	3	LDA $aaaa,Y
			TAX
			TXS

DEM = DEC + CMP

Decrementa il contenuto della memoria in indirizzo. Lo Status Register viene influenzato dalla comparazione (CMP Mem.) fra A e contenuto Memoria in indirizzo dopo l'avvenuto decremento:

Codice	Mnemonico	N° BYTE	Corrisponde a:
C3	DEM ($aa,X)	2	DEC $aa
C7	DEM $aa	2	CMP $aa
CF	DEM $aaaa	3	
D3	DEM ($aa),Y	2	
D7	DEM $aa,X	2	
DB	DEM $aaaa,Y	3	
DF	DEM $aaaa,X	3	

ISB = INC + SBC

Incrementa il contenuto della Memoria in indirizzo. L'acc. A riporta il risultato della sottrazione da A del contenuto della Memoria in indirizzo dopo l'avvenuto incremento:

Codice	Mnemonico	N° BYTE	Corrisponde a:
E3	ISB ($aa,X)	2	INC $aa
E7	ISB $aa	2	SBC $aa
EF	ISB $aaaa	3	
F3	ISB ($aa),Y	2	
F7	ISB $aa,X	2	
FB	ISB $aaaa,Y	3	
FF	ISB $aaaa,X	3	

ANC = A AND Oper. + Carry

Comando immediato. Esegue l'AND fra A e Operando. Il risultato rimane in acc. e viene settato il carry se A risulta Minus:

Codice	Mnemonico	N° BYTE	Corrisponde a:
0B	ANC#	2	AND #$nn
2B	ANC#	2	BPL $xxxx
			SEC

ANL = # AND A + LSR

Comando immediato. Esegue prima un AND fra A e Operando, poi un LSR di A:

Codice	Mnemonico	N° BYTE	Corrisponde a:
4B	ANL #	2	AND #$nn
			LSR

ARO = # AND A + ROR

Comando immediato. Esegue in primo luogo un AND fra A e Operando, quindi un ROR di A:

Codice	Mnemonico	N° BYTE	Corrisponde a:
6B	ARO#	2	AND #$nn
			ROR

ANX = # AND X

Comando immediato. Esegue un AND fra Operando e registro X. Il risultato si trova in accumulatore:

Codice	Mnemonico	N° BYTE	Corrisponde a:
8B	ANX #	2	TXA AND #$nn

ASB = A AND X + SEC + SBC

Comando immediato. Esegue prima A AND X, poi setta il carry e sottrae l'Operando dall'acc. A. Il risultato va nel registro X:

Codice	Mnemonico	N° BYTE	Corrisponde a:
CB	ASB #	2	STX $aa AND $aa PHA SEC SBC #$nn TAX PLA

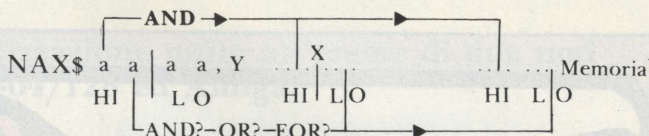
NAY = Nibble AND Y

Esegue un AND fra il nibble HI del registro Y ed il nibble HI della parte HI dell'operando. Si verifica un'operazione non decifrata fra il nibble LO del registro Y ed il nibble LO della parte HI dell'operando. Comunque il risultato delle operazioni viene inviato all'indirizzo indicato in operando:

Codice	Mnemonico	N° BYTE	Corrisponde a:
9C	NAY $aaaa,X	3	Non esiste.

NAX = Nibble AND X

Come sopra, salvo il fatto che il registro interessato dalle operazioni è X:

Codice	Menmonico	N° BYTE	Corrisponde a:
9E	NAX $aaaa,Y	3	Non esiste.

Si è notato che se il Bit 0 di X è ad 1 ed il Bit 0 della parte HI dell'indirizzo dell'operando è a 0, allora il Bit 0 della Memoria assume lo stesso valore del Bit 0 di X.

Il Codice EB è simile al Codice E9, cioè SBC # . ■

La video digitalizzazione

Un'analisi del processo di video-digitalizzazione unito all'esame di due noti digitalizzatori per C-64/128 ed Amiga.

Di Matthew Leeds

Le mie prime esperienze su un computer risalgono a quando utilizzavo un terminale telescrivente senza schermo. L'unico sistema con cui la macchina poteva comunicare con me era attraverso la stampa su carta delle informazioni. In quegli anni infatti era pressochè impossibile procurarsi terminali con uscita video ed i pochi esistenti avevano prezzi esorbitanti. Per visualizzare l'equivalente di una schermata di informazioni era necessario quindi attendere la stampa di venticinque linee di dati. I pochi giochi che si potevano realizzare erano caratterizzati da un'estrema lentezza di esecuzione ed il massimo della creatività fu raggiunto da qualcuno che scrisse un programma che disegnava, utilizzando le lettere O ed X, un noto personaggio dei cartoni animati.

I tempi, naturalmente, sono cambiati. Oggi i monitor video hanno prezzi molto contenuti e non costituiscono più una novità per nessuno. Ma quanti in realtà hanno mai riflettutto sulle differenze che intercorrono fra le modalità di funzionamento di un monitor e quelle di un computer? Fra le onde analogiche del video ed i bit on/off (accesi e spenti, n.d.r.) del mondo digitale esiste una grandissima differenza.

I monitor video per i computer sono stati uno dei risultati più importanti tra quelli legati alla crescita dell'industria televisiva. La produzione di massa dei tubi a raggi catodici (i CRT, n.d.r.), e dei circuiti elettronici che li supportano, hanno determinato un notevole abbassamento dei costi, mentre la guerra tecnologica che si è scatenata per la costruzione di apparecchi TV sempre migliori ha determinato innovazioni tecniche continue.

Fino agli anni '70, però, le ricerche non avevano come obiettivo lo studio di video dedicati specificatamente all'utilizzo nel campo dell'informatica. Solo in quegli anni, infatti, gli ingegneri diedero sviluppo a progetti per la realizzazione di monitor a basso costo adatti a questo scopo.

L'ostacolo più grande era costituito dalla necessità di trasformare i dati presenti nel computer in segnali video. L'intera memoria di un computer, infatti, è costituita da una lunga serie di numeri, 1 e 0, mentre i segnali video si compongono di un'unica onda continua con un movimento sinusoidale, senza però le interruzioni o pause che caratterizzano i bit on/off della memoria di un elaboratore. La soluzione a questo problema fu la creazione dei convertitori digitale/analogico. Attraverso l'ausilio di un particolare circuito si possono infatti eliminare gli intervalli propri del segnale digitale, «riempiendo» gli spazi di tempo inutilizzati in modo da originare un segnale continuo che il monitor sia in grado di interpretare e visualizzare.

In questi ultimi anni, invece, l'evoluzione tecnologica ha fornito i mezzi e gli strumenti atti all'affermarsi e all'espandersi di un nuovo mercato e di nuovi criteri di utilizzo del video in connessione con l'elaboratore.

Gli studi televisivi, gli artisti e moltissime altre persone interessate, fanno uso, da qualche anno, di un processo inverso rispetto a quello appena descritto, che permette cioè di immettere immagini video all'interno della memoria del computer. Tale processo è conosciuto col nome di video-digitalizzazione e la chiave di questo meccanismo è costituita appunto dall'uso dei convertitori analogico/digitale.

Come è facile immaginare, la funzione di questi convertitori consiste nello «spezzare» il segnale video in bit digitali; ci sono, però, altri due importanti parametri che un convertitore A/D deve rispettare nella fase di memorizzazione: la posizione di ogni pixel e la sua relativa luminosità.

La luminosità di ogni pixel

può essere valutata calcolando il voltaggio del segnale video nel momento in cui il pixel stesso viene inviato. Questo valore è quindi confrontato con un valore medio interno prefissato. Varie comparazioni possono creare una scala per la misurazione della luminosità. Ogni valore fa parte della scala che viene poi utilizzata per realizzare l'immagine finale. Per misurare più di un valore limite ci si serve di tracce luminose successive che si ripetono in un'unità di tempo. La posizione di ogni pixel viene calcolata confrontandone la sincronizzazione nel segnale video. I dati che servono a stabilire la temporizzazione orizzontale e verticale dei pixel vengono utilizzati per redigere una bit map dell'immagine video; questi dati vengono poi assemblati con quelli relativi alla luminosità per produrre l'immagine digitalizzata finale.

Questa serie di processi assorbe una percentuale considerevole della potenza di calcolo del computer. I video-digitalizzatori più economici si servono del calcolo di alcune delle variazioni di una traccia luminosa. A seconda del numero delle tonalità di grigio e dei colori presenti nell'immagine finale variano proporzionalmente i tempi di digitalizzazione, che possono superare anche i trenta secondi.

Non appena l'immagine è stata digitalizzata, può essere salvata su disco, inserita in un programma di grafica, modificata, stampata, analizzata, utilizzata come parte di un data base, visualizzata in uno slide show e sfruttata a propria discrezione. Per utilizzare un video-digitalizzatore è necessario servirsi di una fonte di ingresso video come una videocamera o un videoregistratore.

Ho avuto modo di procedere alla prova di due video-digitalizzatori per computer Commodore: Eye-Scan, dedicato al C-64/128, e DigiView per Amiga; di seguito riportiamo le caratteristiche salienti emerse dai test.

Eye-Scan per C-64/128

Si tratta di un pacchetto combinato hardware e software dedicato ai computer C-64/128. L'hardware è costituito da una cartuccia direttamente inseribile nell'apposita porta posteriore dell'elaboratore.

La cartuccia è dotata di un unico jack per l'ingresso video RCA e di due controlli per la regolazione della luminosità e della sincronizzazione del segnale video in entrata.

Il software si serve di menù a scomparsa per la selezione di funzioni tra loro diverse. Sono inoltre richiamabili in qualsiasi momento delle schermate di help relative alle varie funzioni.

I modi di digitalizzazione dell'immagine sono tre: Normal, per le immagini in bianco e nero; 4-Level, che si serve di una scala di quattro tonalità di grigio, e 8-Level, che utilizza otto tonalità di grigio. Successivamente è possibile regolare la sincronizzazione e la luminosità attraverso i controlli presenti sulla cartuccia, oppure direttamente per mezzo del software.

Sono disponibili diversi formati per la registrazione dell'immagine: Koala, Blazing Paddles e Doodle. È disponibile anche l'output su stampante. Inoltre sono supportati i comandi DOS per la formattazione dei dischetti, per richiamare la directory e per cancellare i file. Un'apposita opzione permette la creazione automatica di slide show di tutte le immagini presenti sul disco.

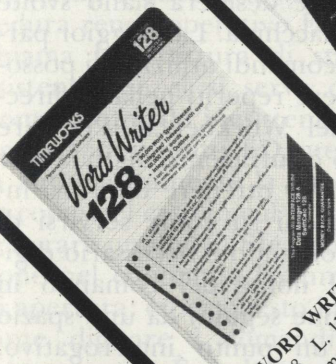
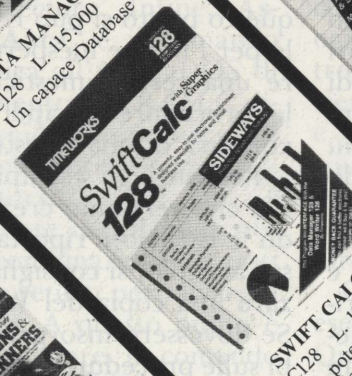
La Digital Engineering, produttrice del package, ha anche provveduto a creare un intero set di utility per programmatori, che consente di disporre di tutte le funzioni supportate dal software di Eye-Scan nella creazione e nello sviluppo dei propri programmi.

DigiView per Amiga

DigiView è un pacchetto hardware/software che fa uso di una videocamera per trasferire immagini a colori e monocromatiche nella memoria dell'Amiga. L'hardware si inserisce direttamente nella porta parallela del computer e contiene un singolo jack video RCA. L'unico altro add-on hardware fornito è un filtro-colore circolare completo di un apposito supporto, che consente la sua istallazione davanti all'obiettivo della videocamera, che permette di separare i colori rosso, verde e blu dell'immagine video. Dal momento che non è possibile utilizzare il filtro unitamente ad un video registratore, non è consentito ottenere immagini a colori da questo tipo di fonte video.

Il software fornito con DigiView è di ottima qualità e si distingue soprattutto per la completezza dei comandi disponibili, attraverso i quali si è in grado di controllare ogni aspetto dell'immagine nelle varie fasi della sua formazione. È possibile infatti regolare la luminosità, il contrasto, la saturazione, l'intensità di ogni componente cromatico e la messa a fuoco dell'immagine stessa, che può essere in seguito registrata come file grafico in standard IFF in 32 colori o come immagine Hold&Modify che si serve di tutti i 4096 colori disponibili. È previsto anche l'immagazzinamento di immagini monocromatiche in alta risoluzione. Un'altra opzione consente la memorizzazione di una serie di dati relativi alle correzioni intervenute nel processo di digitalizzazione, offrendo così all'utente la possibilità di conservare gli effetti delle correzioni anche successivamente al riversamento in un altro standard della figura.

Il software di DigiView è stato recentemente oggetto di revisioni e modifiche. Le novità includono un filtro colore motorizzato (automatico, n.d.r.), la possibilità di cambiare il numero dei colori presenti in un'immagine IFF ed un nuovo sistema di digitalizzazione in grado di migliorare sensibilmente la risoluzione della stessa. È inoltre in fase avanzata di studio un pacchetto software per la creazione di effetti speciali simili a quelli utilizzati nelle produzioni televisive. ■

TIMEWORKS®
NOVITÀ
SIDEWAYS
C64-C128 L. 50.000
Stampa di lato
tuoi Fogli
Elettronici
WORD WRITER
C64 L. 85.000
Un completo
Trattamento Testi
WORD WRITER
IBM-PC e compatibili
L. 135.000
Un completo
Trattamento Testi
WORD WRITER
C128 L. 115.000
Un completo
Trattamento Testi
DATA MANAGER 2
C64 L. 85.000
Un capace Database
DATA MANAGER
IBM-PC e
COMPATIBILI
L. 135.000
Un capace Database
DATA MANAGER
C128 L. 115.000
Un capace Database
PARTNER
IBM-PC e
COMPATIBILI
L. 135.000
Accessori da scrivania
e utility
sempre in linea
PARTNER
C128 L. 120.000
(CARTRIDGE)
Accessori da scrivania
sempre in linea
SWIFT CALC
C128 L. 115.000
Un potente
Foglio Elettronico
con grafici
PRINTWARE
C64-C128 L. 18.900
(PREZZO UNITARIO)
Figure già pronte
per infiniti usi
TOPWARE S.A.S.
Via Gavinana, 17 - 21052 BUSTO ARSIZIO (VA)
Telefono 0331/679219
professional and leading software
Nome:
Cognome:
Indirizzo:
Città:
Per ricevere il nostro kit dimostrativo per il C 128 comprendente un disco e un depliant, ritagliate il tagliando allegato, compilatelo con i vostri dati e speditelo in busta chiusa con L. 4.000 (anche in francobolli) al nostro indirizzo.
TRA BREVE DISPONIBILI
ANCHE PER APPLE
E
ATARI ST
ADERENTE
ASSOFT

Batch File con l'Amiga

Un'introduzione alle potenzialità operative dell'AmigaDOS.

Di Matthew Leeds

La prima versione del manuale-utente dell'Amiga tratta in modo insufficiente la Command Line Interface (CLI, n.d.r.) ed i vari comandi disponibili nella C directory. Non è presente alcuna informazione riguardo alla possibilità di creare batch file e di mandarli in esecuzione utilizzando il comando execute. Nei batch file risiedono invece grandissime potenzialità operative che attendono solo di essere sfruttate; il primo passo in questo senso consiste nell'assimilare le conoscenze indispensabili per poterli creare.

L'ambiente del Workbench formato da icone, mouse e menu a scomparsa è quello più noto alla maggior parte dei possessori di un Amiga. Esso permette di utilizzare l'Amiga più facilmente, con modalità operative più intuitive ed in generale in modo più piacevole. In ogni caso, come ogni sistema grafico, possiede alcune limitazioni: come è ad esempio possibile ottenere ricerche wild card di file o accedere a funzioni per le quali nessun programmatore ha creato un'icona? Fortunatamente i progettisti dell'Amiga hanno fornito all'utente un altro sistema attraverso cui dialogare con la macchina, la CLI, la cui struttura di comandi richiede da una parte un po' più di esperienza, ma permette dall'altra l'accesso ad una potenza operativa e ad un controllo molto superiori.

È possibile accedere al CLI attraverso il Workbench, nel quale è presente un'apposita icona la cui selezione, attraverso la freccia controllata dal mouse, dà luogo all'apertura di una nuova finestra. Nel caso in cui l'icona del CLI non risultasse essere presente nel disco del Workbench è sufficiente controllare se nei parametri del Preferences il CLI è inserito o disinserito (ON o OFF, n.d.r.) ed in quest'ultimo caso è necessario selezionare attraverso la freccetta controllata dal mouse la funzione ON. Una volta ottenuta una finestra CLI, è indispensabile imparare a chiuderla: digitando il comando **ENDCLI** la finestra si chiude istantaneamente. A questo punto si può rivelare utile per l'utente inesperto acquisire una certa dimestichezza con le procedure di apertura e di chiusura di una finestra CLI.

Per tutti gli esempi di batch file che verranno creati seguendo le istruzioni riportate in questo articolo, si consiglia di utilizzare una copia del Workbench. Se dovessero insorgere dei dubbi sulle procedure necessarie alla creazione di una copia del Workbench è sufficiente consultare il manuale-utente dell'Amiga. Prima di proseguire è necessario assicurarsi però che il Workbench si trovi nel drive interno dell'Amiga.

I batch file costituiscono una delle modalità operative più potenti per lo sfruttamento del CLI. Questi file contengono un set di comandi che permette all'utente di programmare le funzioni che desidera siano svolte dalla macchina. La maggior parte dei comandi utilizzabili possono essere reperiti nella C directory del Workbench. Per fare questo è sufficiente digitare **Dir df0:c**. Per ottenere ulteriori informazioni su come servirsi di ogni comando è necessario digitare il nome del comando in questione seguito da uno spazio e da un punto interrogativo, **command ?**. Per chi possiede qualche esperienza di programmazione o ha dimestichezza con sistemi operativi come CP/M o MS-DOS, alcuni di questi comandi potranno risultare familiari.

Nel disco del Workbench è presente un esempio di batch file: è chiamato startup-sequence e si trova nella S: directory. Per prenderne visione, è indispensabile aprire una finestra CLI e digitare **Type s/startup-sequence**. Il file che appare sullo schermo contiene tre comandi differenti. Il primo in ordine di visualizzazione è **Echo**, che scrive sullo schermo ciò che è contenuto tra le virgolette. Il secondo è **LoadWb**, cui è preposto il caricamento del Workbench. Il terzo è costituito da una combinazione di due comandi: **Endcli** ed un nuovo indirizzamento a **Nil:**. Questa combinazione chiude la finestra CLI corrente, aperta al caricamento del Workbench, ed invia il messaggio di fine alla device **Nil:**, che riceve il messaggio da ciò che è stato scritto sullo schermo.

La sequenza di avvio descritta viene eseguita ad ogni accensione dell'elaboratore. Operando delle modifiche è possibile alterare le procedure iniziale che interessano l'Amiga al momento della sua accensione. Se si desidera modificare la sequenza di accensione (startup-sequence, n.d.r.), è possibile modificare il file, utilizzando un word processor che possa creare file ASCII, o usare ED, l'editore di testo incluso nel Workbench. Per redigere la sequenza di accensione utilizzando ED, da una finestra CLI, è necessario digitare **ED s/startup-sequence**. Questa procedura rende operativo ED e richiama la struttura di startup esistente. Se ci si serve di un nome di un file inesistente o di un path inesistente, ED interpreta automaticamente la procedura come nel caso in cui si volesse generare un nuovo file, svolgendone nel contempo la funzione di apertura; è per questo importante digitare i comandi con estrema cura. È inoltre necessario ricordare che per l'AmigaDOS non ha importanza se le lettere sono digitate maiuscole o minuscole.

Normalmente ED si trova nel modo inserimento; per il movimento del cursore è sufficiente utilizzare gli appositi tasti e, premendo return alla fine di una linea, si accede ad una nuova riga. La medesima funzione è ottenibile anche combinando la pressione del tasto control con A. Può rivelarsi degno di interesse aggiungere qualche messaggio personale o il comando **Date on** su una linea separata per ottenere la visualizzazione dell'ultima data di accesso al file presente sul disco. Una volta che la procedura di editazione del file è terminata, bisogna premere il tasto di escape, X e return, impostando così la scrittura del file sul disco. Se non si desidera registrare su disco i cambiamenti operati è sufficiente servirsi della combinazione escape, Q e return, consentendo l'uscita di ED senza la scrittura su disco.

Creare dei propri batch file, utilizzando ED, è molto semplice. Una raccomandazione utile può essere comunque quella di posizionare tutti i batch file nella S: directory. Quando infatti viene utilizzato il comando execute per dare il run ai batch file, l'AmigaDOS ricerca per prima cosa il file nella directory corrente e, nel caso non lo individui, sposta la sua attenzione direttamante alla S: directory. Inserendo i batch file nella S: directory non si ha così la necessità di doversi preoccupare di quale sia la directory corrente.

Per aprire un batch file chiamato RAMDISK che crei un ram disk contenete la C: directory è sufficiente digitare **ED s/RAMDISK**. Dal momento che nella S: directory non è presente alcun file chiamato RAMDISK, ED interpreta attraverso tale istruzione l'intenzione da parte dell'utente di crearne uno nuovo. Sono sufficienti tre comandi:

```
Makedir RAM:C
Copy SYS:C RAM:C All Quiet
Assign C: RAM:C
```

Makedir RAM:C crea nella RAM una directory chiamata C:. **Copy SYS:C RAM:C All Quiet** copia la system:C directory nella RAM:C directory. Infine, **Assign C: RAM:C** è il comando che comunica all'AmigaDOS di fare riferimento alla RAM:C directory ogniqualvolta necessiti di un comando proveniente dalla C: directory.

È importante ricordarsi di utilizzare la combinazione di tasti escape, X e return per salvare il file su disco. A questo punto è possibile utilizzare il comando **Execute RAMDISK** per creare una C: directory nella RAM e rimuovere il disco del Workbench dal drive. Se non si desidera digitare integralmente il comando Execute è possibile assegnargli un nome differente e più corto, sfruttando l'istruzione Rename, battezzandolo per esempio DO o GO. La sintassi per l'esecuzione di questa procedura è la seguente: **Rename C/Execute to C/GO**, che, aggiunta come ultima riga nel file RAMDISK, consente poi di digitare semplicemente **GO RAMDISK**.

L'uso dei batch file permette una considerevole varietà di applicazioni. È per esempio possibile modificare il comando Rename o creare un batch file che potenzi Rename. In questo file si possono usare alcuni nuovi comandi, non presenti nella C: directory, ma comunque sfruttabili nei batch file. **Key** consente di impostare parametri, permettendo all'utente di creare le strutture per il command file. **If** e **Not** sono i condizionali logici standard. **Else** imposta il comando che deve essere eseguito nel caso in cui il testo logico fallisca. **Endif** decreta quindi la fine del loop logico. La struttura del file è la seguente:

```
.Key from,to
IF NOT EXISTS <to>
RENAME <from> AS <to>
ELSE
ECHO "<to> esiste già. Prova un altro nome."
```

Un altro semplice, ma utile, file è il seguente:

```
CD DFI:
DIR OPT A
LIST
DIR > PRT: OPT A
```

Il file sopra riportato può essere denominato OUT (da check out: controlla, n.d.r.): ogni volta che si desidera controllare un nuovo disco è sufficiente inserirlo nel drive esterno e digitare **Execute OUT**. La routine presenta sullo schermo una directory estesa, esegue il LIST ed invia il contenuto della directory estesa alla stampante.

Ci sono molte altre operazioni eseguibili attraverso i batch file. Questo articolo non intende costituire una guida completa per l'uso di ED o per la creazione dei file stessi, bensì una panoramica sulle incredibili potenzialità dell'AmigaDOS.

Per chi desiderasse approfondire lo studio di questi comandi è consigliato l'acquisto del volume «Il manuale dell'AmigaDOS» pubblicato dalla IHT (la casa editrice di Commodore Gazette, n.d.r.) disponibile a partire dal mese di giugno. ■

VI
LEZIONE

CORSO DI PROGRAMMAZIONE

Impariamo a programmare in linguaggio macchina il Commodore 64.

Sesta parte

Di Sergio Fiorentini

Attenzione: nel corso dell'articolo vengono talvolta utilizzate lettere maiuscole in modo sintatticamente improprio, questo avviene per particolari esigenze didattiche. Dal momento che le diverse puntate, che costituiscono questo corso, si integrano reciprocamente, consigliamo a quanti ci seguissero per la prima volta di richiedere le copie arretrate della rivista (i numeri 1-2-3-4/86, 1/87).

Nelle scorse puntate sono state esaminate le istruzioni aritmetico-logiche: ADC, SBC, CMP, AND, ORA ed EOR, che hanno la particolarità di eseguire un'operazione fra 2 operandi, rispettivamente contenuti nell'Accumulatore del Microprocessore ed in un generico registro di memoria, constatando che il risultato di tali operazioni sostituisce nell'Accumulatore il primo operando. Il Microprocessore 6502 (analogamente al 6510) dispone inoltre di un'altra serie di operazioni aritmetico-logiche, che necessitano, a differenza delle precedenti, di un solo operando. Queste istruzioni, che prenderemo singolarmente in esame, potranno agire o sull'Accumulatore oppure su un registro di memoria. Il risultato di tali operazioni verrà comunque riposto nello stesso registro di provenienza dell'operando (ovvero nell'Accumulatore o in un registro di memoria).

La prima di tali istruzioni è denominata ASL (Aritmetic Shift Left: scorrimento aritmetico a sinistra) e consente di far «scorrere» il contenuto di un byte, di un bit, verso sinistra: nel bit meno significativo, bit(0), verrà introdotto uno %0, il precedente bit(0) verrà «fatto slittare» nel bit(1), mentre il contenuto di questo bit troverà posto nel bit(2) e così via per tutti i rimanenti bit presenti nel byte. Il bit più significativo, bit(7), «cadrà» nel bit del riporto (C) del registro di Stato (SR). L'effetto dello scorrimento a sinistra di tutti i bit di un byte è quello di moltiplicare per 2 il valore di tale byte: da qui il nome di scorrimento aritmetico con il quale si indica questa istruzione. «ASL» imposta anche il bit di segno (N), nel Registro di Stato (SR), allo stesso valore del bit più significativo del byte, dopo lo slittamento, di modo che anche il bit di zero (Z), impostato ad %1 o ad %0, venga comunque inserito nel Registro di Stato, a seconda che il byte, al termine dell'operazione, risulti o meno essere nullo ($00). L'istruzione dispone di diversi modi d'indirizzamento:

implicito: il codice Mnemonico non è seguito da nessun operando ed opera sull'Accumulatore in 2 cicli macchina necessari rispettivamente per la lettura e l'esecuzione del codice e dello scorrimento

diretto: il codice Mnemonico precede l'indirizzo della locazione dalla quale prelevare l'operando prescelto per l'operazione di slittamento. In questo caso l'esecuzione dell'istruzione necessita, per poter essere portata a termine, di 6 cicli così strutturati: tre di lettura, per leggere l'istruzione ed il relativo indirizzo dell'operando nella memoria programma, un'ulteriore ciclo, sempre di lettura, per acquisire l'operando dalla memoria dati, un ciclo per operare lo slittamento del byte all'interno dell'Unità Aritmetico-Logica (ALU) ed infine un ultimo ciclo di scrittura, per riposizionare il byte nella stessa locazione dalla quale era stato prelevato

diretto in pagina zero: sostanzialmente simile

al precedente modo d'indirizzamento dal quale differisce per il fatto che la locazione di memoria da cui reperire l'operando risiede nella pagina zero e che necessita, per poter essere eseguita, di 5 cicli macchina. Volendo, a titolo di esempio, moltiplicare per 2 il contenuto della locazione $20000, si potranno utilizzare indifferentemente i 2 programmi seguenti, ubicati rispettivamente agli indirizzi $1000 e $1010.

```
.A    1000    ASL    $2000
.A    1003    BRK

.A    1010    LDA    $2000
.A    1013    ASL
.A    1014    STA    $2000
.A    1017    BRK
```

I due programmi trovano sostanzialmente il medesimo impiego e necessitano ognuno di 10 cicli macchina per la loro esecuzione. Volendo provare a moltiplicare per 2 il numero $53, oel bit(2) e così via per tutti i rimanenti bit presenti nel byte. Il bit più significativo, bit(7), «cadrà» nel bit del riporto (C) del registro di Stato (SR). L'effetto dello scorrimento a sinistra di tutti i bit di un byte è quello di moltiplicare per 2 il valore di tale byte: da qui il nome di scorrimento aritmetico con il quale si indica questa istruzione. «ASL» imposta anche il bit di segno (N), nel Registro di Stato (SR), allo stesso valore del bit più significativoempre un byte. Se questa condizione non viene rispettata il bit più significativo del risultato, impostato ad %1, «cade» inevitabilmente nel bit del riporto (C) del registro di Stato (SR).

Per eseguire operazioni di raddoppio in precisione multipla, ovvero di numeri che occupino più byte consecutivi, si rende indispensabile l'utilizzo di una nuova istruzione: ROL (ROtation Left: rotazione a sinistra), che differisce dalla precedente per il fatto che, preliminarmente all'operazione stessa, anziché inserire uno %0 nel bit(0) del registro da ruotare, viene inserito il precedente valore del bit del riporto (C). Utilizzando quest'ultima istruzione, dopo un ASL è possibile continuare nel byte più significativo l'operazione di traslazione, dal momento che l'istruzione ASL ripone il bit più significativo, slittato, nel bit del riporto, (C), dal quale viene poi prelevato dall'istruzione ROL per portare a termine, nel byte successivo, l'operazione di traslazione. I modi di indirizzamento disponibili per l'istruzione ROL sono gli stessi dell'istruzione ASL (Implicito, Diretto, Diretto in Pagina Zero); risulta essere identico anche il modo in cui vengono impostati i bit di Segno e di Zero, (N) e (Z), nel Registro di Stato (SR) a seconda del risultato dell'operazione.

Dal punto di vista pratico l'istruzione ASL può essere sostituita all'interno di un programma dall'utilizzo dell'istruzione ROL preceduta a sua volta dall'istruzione CLC.

Presentiamo ora tre programmi che permettono di moltiplicare per 2 il numero, nel formato di 2 byte, contenuto nelle locazioni $2000 e $2001, rispettivamente per il byte basso ed alto.

```
.A    1000    ASL    $2000
.A    1003    ROL    $2001
.A    1006    BRK

.A    1010    LDA    $2000
.A    1013    ASL
.A    1014    STA    $2000
.A    1017    ROL    $2001
.A    101A    BRK

.A    1020    ASL    $2000
.A    1023    LDA    $2001
.A    1026    ROL
.A    1027    STA    $2001
.A    102A    BRK
```

Dopo aver inserito il numero da moltiplicare nei byte $2000 e $2001, mediante il comando Interrogate (I), è sufficiente lanciare uno dei tre programmi sopraindicati; al proposito è consentito utilizzare i comandi G o W del monitor, seguiti dall'indirizzo della prima istruzione di detti programmi.

Servendosi delle istruzioni ASL e ROL, unitamente a CLC e ADC, è possible implementare anche moltiplicazioni per 3, 4, 5, 6, ecc. Volendo, per esempio, moltiplicare per 6 un numero in doppia precisione, che occupa cioè 2 byte, è necessario moltiplicare per due detto numero, addizionare il primo numero al prodotto, ottenendo così una moltiplicazione per tre, e successivamente rimoltiplicare l'ultima somma nuovamente per due. Infatti: 6 = (2+1)*2.

Vi presentiamo dunque, a partire dalla locazione $1000, il programma che consente l'esecuzione di tale moltiplicazione. Il numero da moltiplicare per 6 deve essere locato, nel formato byte basso, byte alto, alle locazioni $2000, $2001. Il risultato viene riposto, al termine del programma, sempre nello stesso formato, nei byte $2002, $2003.

```
.A    1000    LDA    $2001
.A    1003    STA    $2003
.A    1006    LDA    $2000
.A    1009    ASL
.A    100A    ROL    $2003
.A    100D    ADC    $2000
.A    1010    STA    $2002
.A    1013    LDA    $2003
.A    1016    ADC    $2001
.A    1019    ASL    $2002
.A    101C    ROL
.A    101D    STA    $2003
.A    1020    BRK
```

Se facendo «slittare» un numero di un bit verso sinistra si esegue una moltiplicazione per due, analogamente facendo «slittare» il numero verso destra lo si divide per due. All'uopo è stata prevista l'istruzione LSR (Logic Shift Right: scorrimento logico a destra), istruzione che, operando sull'Accumulatore o su un registro di memoria, attua lo scorrimento verso destra di un bit: nel bit più significativo, bit(7), viene inserito un %0, mentre ciò che in precedenza era contenuto in questo bit viene im-

magazzinato nuovamente nel bit(6), che ha a sua volta ceduto i suoi dati al bit(5), e così via procedendo verso destra. Il contenuto del bit meno significativo, bit(0), viene trasferito nel bit di riporto (C) del Registro di Stato (SR).

Analogamente alle istruzioni di traslazione e rotazione a sinistra (ASL e ROL), i bit di zero (Z) e di segno (N) vengono impostati a seconda del risultato della traslazione e, dal momento che il bit(7) viene impostato a %0, lo stesso valore viene copiato nel bit di segno (N).

Le forme di indirizzamento, la lunghezza delle istruzioni in codice macchina ed i tempi di esecuzione sono i medesimi anche nell' analoga istruzione di scorrimento a sinistra.

L'istruzione LSR è considerata un'istruzione logica, in quanto consente la divisione per due dei soli numeri assoluti e di quelli relativi (rappresentati in complemento a due) positivi. I numeri relativi negativi, fatti slittare a destra, a causa dell'introduzione di un %0 nel bit più significativo, divengono positivi. Illustreremo in seguito le operazioni necessarie per la loro divisione.

Dal momento che il bit meno significativo (bit(0)) segnala se un numero è pari o dispari, a seconda che sia impostato a %0 o a %1, in entrambi i casi, al termine dell'esecuzione dell'istruzione LSR, tale bit viene trasferito nel bit di riporto (C) al fine di indicare un eventuale resto della divisione per due.

Riportiamo qui di seguito due programmi, ubicati agli indirizzi $1000 e $1010, che consentono la divisione per due del contenuto della locazione $2000 e che risultano equivalenti tra loro nella funzione svolta:

```
.A   1000   LSR   $2000
.A   1003   BRK

.A   1010   LDA   $2000
.A   1013   LSR
.A   1014   STA   $2000
.A   1017   BRK
```

Al fine di consentire le divisioni in precisione multipla è stata predisposta l'istruzione ROR (ROtation Right: rotazione a destra), che differisce dalla LSR per il fatto che nel bit più significativo viene inserito il precedente valore del bit del riporto (C) del Registro di Stato (SR). Le forme di indirizzamento, i tempi di esecuzione ed il modo in cui vengono impostati i bit di segno (N) e di Zero (Z) nel Registro di Stato sono analoghi a quelli della ROL.

Sulla base di quanto affermato, l'istruzione LSR può quindi essere sostituita da una ROR, preceduta da una CLC.

Nelle divisioni per due a precisione multipla si inizia con lo slittamento verso destra del byte più alto (con una LSR), procedendo poi con l'utilizzo delle ROR per tutti i rimanenti byte, onde consentire di rientrare nei byte di ordine più basso ai bit uscenti a destra da ogni byte, tramite il registro del riporto (C). Vengono ora riprodotti due esempi di divisione in doppia preisione che operano sul numero contenuto alle locazioni $FB e $FC in Pagina Zero.

```
.A   1000   LSR   $FC
.A   1002   ROR   $FB
.A   1004   BRK

.A   1010   LSR   $FC
.A   1012   LDA   $FB
.A   1014   ROR
.A   1015   STA   $FB
.A   1017   BRK
```

Qualora il numero da dividere sia un numero relativo, espresso in complemento a due, si rende indispensabile inserire nel bit più significativo del byte più significativo un bit dello stesso valore di quello precedente fatto slittare verso destra, al fine di mantenere uguaglianza di segni fra dividendo quoziente. Il programma seguente, grazie ad un particolare accorgimento, consente la divisione per due di un numero relativo occupante 2 byte. Il programma ha inizio alla locazione $1000, mentre il numero da dividere deve essere locato agli indirizzi: $2000 e $2001.

```
.A   1000   LDA   $2001
.A   1003   ASL
.A   1004   ROR   $2001
.A   1007   ROR   $2000
.A   100A   BRK
```

Esistono, nell'assembler del 6510, altre due istruzioni che operano con un solo operando: INC (INCrement: incrementa), che incrementa il contenuto di un generico registro di memoria, e DEC (DECrement), che ha funzioni esattamente opposte. Entrambe le istruzioni agiscono solo su Registri Esterni, avendo a disposizione forme di indirizzamento Dirette e Dirette in Pagina Zero. Non sono disponibili forme d'indirizzamento Implicito, ma, d'altro canto, l'incremento e il decremento dell'Accumulatore possono essere facilmente ottenuti grazie alle seguenti successioni di istruzioni:

```
CLC          e     SEC
ADC # $01          SBC # $01
```

L'istruzione INC (e per conseguenza diretta anche DEC) preleva e ripone l'operando nella stessa locazione dopo averlo incrementato o decrementato, necessitando, nelle forme di indirizzamento Diretto e Diretto in Pagina Zero, per poter essere eseguita, rispettivamente di 5 e 6 cicli macchina, che vengono utilizzati con le medesime modalità seguite per le istruzioni ASL, LSR, ROL e ROR. Al termine delle operazioni vengono impostati i bit di segno (N) e di zero (Z), a seconda del segno e dell'annullamento del risultato. Contrariamente ad altre istruzioni aritmetiche, il bit del riporto (C) non viene minimamente influenzato dal risultato.

Vengono di seguito riportati due elementari

programmi che consentono rispettivamente di incrementare e decrementare il contenuto della locazione $2000.

```
.A      1000      INC      $2000
.A      1003      BRK

.A      1010      DEC      $2000
.A      1013      BRK
```

Le istruzioni esaminate sino a questo punto trovano utilizzo nell'eseguire trasferimenti ed operazioni aritmetico/logiche sui dati e nell' impostare alcuni bit (C, V) nel Registro di Stato. Dette istruzioni seguono una procedura di esecuzione sequenziale, nello stesso ordine in cui sono cioè caricate nella memoria programma. Compito del Program Counter (PC: contatore programma) è proprio quello di memorizzare l'indirizzo del primo byte (Operational Code: codice operativo) della prossima istruzione in ordine di esecuzione. Tale contatore viene incrementato automaticamente ad ogni istruzione, per procedere di pari passo con il programma elaborato. In molti programmi è tuttavia necessario rompere la sequenzialità delle istruzioni, seguendo delle «Diramazioni» ovvero dei «salti» ad altri segmenti di programma, a seconda che si verifichino determinate «condizioni».

Come si è constatato, il Registro di Stato (SR) contiene alcuni bit (bit del riporto, di Zero, di overflow e di segno: C, Z, V, N) che vengono modificati a seconda dei byte trasferiti, o dei risultati delle operazioni aritmetico/logiche eseguite dall'ALU. I quattro bit sopra citati ricoprono questa posizione all'interno del Registro di Stato (SR):

```
bit(7), (6), (5), (4), (3), (2), (1), (0)
(N),(V),(.), (.), (.), (.), (Z), (C)
```

È possibile redigere una tabella che indichi le modificazioni di questi quattro bit nelle varie istruzioni sin qui esaminate. Nella tabella i bit non importanti sono sostituiti con un punto (.), i bit che vengono settati a $0 con uno 0, i bit impostati a $1 con un 1, i bit modificati a seconda dei risultati dell'operazione sono indicati dalla lettera M, mentre quelli non influenzati con la lettera X.

```
Istru.   N,V,.,.,.,.,Z,C
CLC      X,X,.,.,.,.,X,0
SEC      X,X,.,.,.,.,X,1
CLV      X,0,.,.,.,.,X,X

LDA      M,X,.,.,.,.,M,X
STA      X,X,.,.,.,.,X,X

ADC      M,M,.,.,.,.,M,M
SBC      M,M,.,.,.,.,M,M
CMP      M,X,.,.,.,.,M,M

AND      M,X,.,.,.,.,M,X
ORA      M,X,.,.,.,.,M,X
EOR      M,X,.,.,.,.,M,X

ASL      M,X,.,.,.,.,M,M
LSR      0,X,.,.,.,.,M,M
ROL      M,X,.,.,.,.,M,M
ROR      M,X,.,.,.,.,M,M
INC      M,X,.,.,.,.,M,X
DEC      M,X,.,.,.,.,M,X
```

Nel riassumere per sommi capi il contenuto della tabella, si può dire che le istruzioni di trasferimento dalla MPU verso l'esterno (STA) lasciano inalterati tutti i bit del Registro di Stato (SR), le istruzioni di trasferimento dati dall'MPU all'esterno (LDA) e le istruzioni logiche (AND, ORA, EOR) modificano i bit di Zero e quello di segno. Le istruzioni aritmetiche (ADC, SBC, CMP) moficiano tutti i quattro bit del Registro di Stato, ad eccezione della CMP che lascia invariato il bit di overflow (V). Le istruzioni di scorrimento (SL, LSR, ROL, ROR) trasformano i bit di riporto, zero e segno. Le istruzioni di conteggio 8INC e DEC) modificano i bit di zero e di segno. Le istruzioni di controllo sul Registro di Stato (CLC, SEC, CLV) agiscono esclusivamente sui bit ai quali si riferiscono, lasciando inalterati gli altri. Inoltre, per tutte le istruzioni, il modo in cui vengono impostati i bit nel Registro di Stato è indipendente dai particolari modi di indirizzamento utilizzati nell'istruzione.

Le istruzioni che verranno esaminate fra poco consentono di interrompere il flusso sequenziale di un programma, a seconda che siano impostati o meno determinati bit nel Registro di Stato (SR). Poiché l'impostazione di tali bit, come si è appena illustrato, dipende dai risultati delle varie operazioni eseguite dalla MPU, è possibile, mediante tali istruzioni di diramazione condizionata, far eseguire al Microprocessore delle «scelte».

Le istruzioni di diramazione sono le seguenti:

BCC (Branch Carry Clear: dirama se riporto nullo), che fa si che il flusso del programma «salti» ad un nuovo indirizzo (specificato dall'operando dell'istruzione), nel caso il bit di riporto (C), nel Registro di Stato (SR), sia impostato a %0. In caso contrario l'esecuzione del programma continua dall'istruzione seguente quella di diramazione.

BSC (Branch Carry Set: dirama se il riporto è a %1) è un' istruzione contraria alla precedente che consente la diramazione nel caso il bit del riporto (C) sia impostato a %1.

BEQ (Branch EQual: dirama se uguale) è un'istruzione che determina una diramazione nel caso il bit di zero (Z), nel Registro di Stato (SR), sia impostato a %1. La denominazione di questa istruzione deriva dal fatto che se viene fatta seguire un'istruzione di confronto (CMP o anche SBC ed EOR), essa è in grado di eseguire la diramazione nel caso in cui i due operandi confrontati siano uguali. Ovviamente, se il bit di Zero (Z) viene impostato a %0, la diramazione non ha luogo.

BNE (Branch No Egual: Dirama se non uguale) è l'istruzione contraria alla precedente. La diramazione viene eseguita nel caso in cui il bit di

zero (Z) risulti essere impostato a %0.

BVC (Branch oVerflow Clear: dirama se l'overflow è nullo) determina una diramazione nel caso in cui il bit di overflow (V) nel Registro di Stato (SR) sia impostato a %0.

BVS (Branch oVerflow Set: dirama se l'overflow è impostato a %1) è l'istruzione contraria alla precedente.

BLP (Branch PLus: dirama se positivo) determina una diramazione nel caso in cui il bit di segno (N), nel Registro di Stato (SR), risulti essere impostato a %0.

BMI (Branch MInus: dirama se negativo) è l'istruzione contraria alla precedente che determina una diramazione nel caso in cui il bit di segno (N) sia settato a %1.

Le istruzioni di diramazione, una volta eseguite, indipendentemente dal fatto che determinino un «salto» ad altro segmento di programma, lasciano inalterati i bit nel Registro di Stato (SR).

Tutte le istruzioni di diramazione, una volta assemblate, generano 2 byte di codice macchina: uno dedicato al Codice Operativo e l'altro all'operando. Detto operando è un numero binario, relativo, rappresentato in complemento a due, che può essere quindi anche negativo, onde consentire «salti» all'indietro. Nel caso le condizioni siano assolte e la diramazione venga eseguita, detto operando viene sommato algebricamente al contenuto del contatore di programma (PC), facendo sì che esso «punti» all'istruzione successiva. Secondo quanto affermato si deduce che i salti possono essere compresi fra 127 byte in avanti e 128 indietro a partire dal valore corrente del Contatore di Programma (PC), ovvero dall'indirizzo della prima istruzione successiva all'istruzione di diramazione. È per questo motivo che la forma di indirizzamento utilizzata nelle istruzioni di diramazione è detta Relativa, dal momento che il valore della locazione a cui è indirizzata è determinata «in relazione» al contenuto del Contatore di Programma all'atto stesso di eseguire detta diramazione, aggiungendo ad esso il contenuto (che può anche essere un numero negativo) dell'operando dell'istruzione di diramazione, che per questo motivo viene indicato con il nome di «offset» (spiazzamento). Tuttavia, poiché sarebbe laborioso per l'utente calcolare per ogni diramazione il valore dello spiazzamento (o indirizzo relativo), operando la differenza algebrica fra l'indirizzo della locazione alla quale si desidera eseguire la diramazione e l'indirizzo della prima istruzione dopo quella di diramazione, tale differenza, in fase di assemblaggio, viene calcolata automaticamente dal Monitor o Assemblatore al quale occorre comunicare soltanto l'indirizzo «Assoluto» della locazione alla quale eseguire la diramazione. A titolo di esempio il programma seguente determina, nel caso il bit di Zero nel Registro di Stato risulti essere impostato a %1, una diramazione tale da saltare l'istruzione STA che segue quella di diramazione BEQ:

```
.A      1000            BEQ     $1005
.A      1002            STA     $2000
.A      1005            BRK
```

Una volta assemblata l'istruzione, il monitor indica automaticamente il valore dell'offset, in questo caso $03, valore individuabile graficamente dopo il codice operativo dell'istruzione BEQ $F0:

```
.A      1000    F0      03      BEQ     $1005
.A      1002    8D      00 20   STA     $2000
.A      1005    00              BRK
```

Se il valore dello spiazzamento non fosse stato compreso nel campo di un byte relativo (da -128 $80 a +127 $7F), il monitor non avrebbe assemblato la linea comunicando una condizione d'errore.

Per quanto riguarda il tempo di esecuzione, occorre distinguere fra tre casi:

1) La condizione di diramazione non è assolta: in questo caso vengono utilizzati due cicli macchina, entrambi di lettura nella Memoria Programma, al fine di reperire il valore del Codice Operativo e quello dello Spiazzamento; indi si passa, al terzo ciclo, alla lettura del Codice Operativo dell'istruzione successiva a quella di diramazione.

2) La condizione di diramazione è assolta e l'indirizzo di salto è nella stessa pagina di quello dell'istruzione successiva a quella di diramazione: in questo caso occorrono in tutto tre cicli, di cui due, usati analogamente al caso precedente, per leggere Codice Operativo e Spiazzamento, ed un ulteriore ciclo per addizionare algebricamente il valore dello Spiazzamento al byte di ordine basso del Contatore Programma (PC), di modo che questo «Punti» alla locazione contenente il Codice Operativo dell'istruzione a cui si desidera avvenga la diramazione.

3) La condizione di diramazione è verificata e la locazione alla quale eseguire la diramazione non si trova nella stessa pagina della locazione dell'istruzione successiva a quella di diramazione: in questo caso occorrono complessivamente quattro cicli macchina, per portare a termine l'esecuzione dell'istruzione, tre dei quali sono utilizzati analogamente al caso precedente ed un ulteriore ciclo serve per incrementare o decrementare il byte alto del Contatore Programma (PC).

Le seguenti tre coppie di istruzioni, «forzando» alcuni bit nel Registro di Stato (SR), consentono di eseguire delle diramazioni incondizionate:

```
CLC             SEC             CLV
BCC             BCS             BVC
```

I programmi che seguono esemplificano alcuni degli usi più frequenti delle istruzioni di diramazione.

```
.A      1000            LDA     $2000
.A      1003            CMP     $2001
```

```
.A    1006   BCC   $1012
.A    1008   BEQ   $100E
.A    100A   STA   $2002
.A    100D   BRK
.A    100E   STA   $2003
.A    1011   BRK
.A    1012   STA   $2004
.A    1015   BRK
```

Il programma riportato confronta il contenuto della locazione $2000 con quello della locazione $2001 e lo ripone, se questo dovesse essere minore del secondo, alla locazione $2004, se dovesse essere uguale, alla locazione $2003, se dovesse essere maggiore, alla locazione $2002. È importante notare come l'istruzione CMP imposti i bit (C) e (Z) nel Registro di Stato e come questi siano successivamente intercettati dalle varie istruzioni di diramazione.

Il programma che segue confronta invece due numeri in doppia precisione, presenti rispettivamente ai byte $2000, $2001 e $2002, $2003, eseguendo, se il primo è minore, uguale o maggiore del secondo, diramazioni agli indirizzi: $1020, $1014, $1030. Si ponga attenzione a come vengano confrontati prima i byte bassi e, nel caso di eguaglianza, quelli alti.

```
.A    1000   LDA   $2001
.A    1003   CMP   $2003
.A    1006   BCC   $1020
.A    1008   BNE   $1030

.A    100A   LDA   $2000
.A    100D   CMP   $2002
.A    1010   BCC   $1020
.A    1012   BNE   $1030
.A    1014   BRK
```

Il programma seguente determina una diramazione nel caso in cui il bit(4), locato a $00FB, sia impostato a %0; si sottolinei l'esecuzione di una AND con il byte di maschera $10 al fine di azzerare tutti i bit ad eccezione di quello che si intende testare.

```
.A    1000   LDA   $FB
.A    1002   AND   # $10
.A    1004   BEQ   $1010
.A    1006   BRK
```

Se al posto dell'istruzione BEQ ci fosse stata una BNE la diramazione sarebbe ovviamente avvenuta se il bit(4), nella locazione $FB, fosse stato impostato a %1.

Volendo sommare o sottrarre in modo «Immediato» ad un numero in doppia precisione, memorizzato dalla locazione $2000, il numero 40 ($28), si possono utilizzare con indubbi vantaggi i due programmi che seguono:

```
.A    1000   LDA   $2000
.A    1003   CLC
.A    1004   ADC   # $28
.A    1006   STA   $2000
.A    1009   BCC   $100E
.A    100B   INC   $2001
.A    100E   BRK
.A    1000   LDA   $2000
.A    1003   SEC
.A    1004   SBC   # $28
.A    1006   STA   $2000
.A    1009   BCS   $100E
.A    100B   DEC   $2001
.A    100E   BRK
```

Volendo incrementare un numero in doppia precisione allocato a partire dall'indirizzo $2000, si può utilizzare il seguente programma:

```
.A    1000   INC   $2000
.A    1003   BNE   $1008
.A    1005   INC   $2001
.A    1008   BRK
```

Volendo invece calcolare il valore assoluto del numero relativo, rappresentato in complemento a due, allocato alla locazione $2000, si può utilizzare questo programma, che, nel caso il numero risulti essere negativo, ne calcola il complemento logico e lo incrementa.

```
.A    1000   LDA   $2000
.A    1003   BPL   $100D
.A    1005   EOR   # $FF
.A    1007   CLC
.A    1008   ADC   # $01
.A    100A   STA   $2000
.A    100D   BRK
```

Dal momento che le diramazioni possono anche essere effettuate all'indietro, ovvero, in riferimento a istruzioni precedenti quella di diramazione, si possono realizzare delle «Iterazioni», facendo sì che il flusso logico di un programma passi più volte per un certo insieme di istruzioni. Volendo, ad esempio, implementare la moltiplicazione dei due numeri memorizzati rispettivamente nelle locazioni $2000 e $2001, è possibile servirsi del seguente programma, che somma tante volte il moltiplicando a se stesso quante sono quelle specificate dal moltiplicatore. Il risultato viene riposto, nel formato byte basso, byte alto, nelle locazioni: $2002, $2003.

```
.A    1000   LDA   # $00
.A    1002   STA   $2002
.A    1005   STA   $2003
.A    1008   LDA   $2000
.A    100B   BEQ   $1020
.A    100D   CLC
.A    100E   LDA   $2002
.A    1010   ADC   $2001
.A    1013   STA   $2002
.A    1016   BCC   $101B
.A    1018   INC   $2003
.A    101B   DEC   $2000
.A    101E   BNE   $100D
.A    1020   BRK
```

■

La produzione di The Bard's Tale

Intervista in esclusiva a Brian Fargo, presidente della Interplay, e a Joe Ybarra, produttore della Electronic Arts.

Di Matthew Leeds

Il primo gioco che ho avuto modo di vedere su un computer è stato un «Fantasy Role Playing» (FRP, n.d.r.), dotato di quattro livelli di cunicoli e labirinti tra i quali avventurarsi alla ricerca di tesori, armi, formule magiche, lottando contro creature terrificanti. In relazione alla sua soluzione, ho avuto bisogno di più di trenta ore, e mi sembrava un tempo lunghissimo, solo per disegnare una mappa precisa del primo livello che mostrasse ogni passaggio ed ogni curva del labirinto, l'esatta locazione dei tesori, i mostri, le sale in cui li si incontrava ed il modo migliore per sconfiggerli. È stato solo molto tempo dopo che ho avuto modo di rendermi conto della ben più grande quantità di lavoro spesa nella realizzazione di quel gioco, nel disegnare i labirinti, nell'immaginare creature mostruose e nel creare all'interno dell'azione un contesto plausibile in cui inserirle.

Da allora i giochi FRP sono divenuti molto più complessi, i mondi immaginati più ricchi di dettagli, le regole secondo cui operare molto più difficili da comprendere e la grafica più realistica. Uno degli FRP di maggior successo è certamente «The Bard's Tale», prodotto dalla Interplay Productions per la Electronic Arts.

Di recente è stata presentata sul mercato una nuova versione di questo fantastico gioco: «The Bard's Tale II: The Destiny Knight». Per disporre di più approfondite conoscenze sugli FRP ho ritenuto opportuno avere uno scambio di opinioni con Brian Fargo, presidente della Interplay, e con Joe Ybarra, produttore responsabile dell'Electronic Arts per la pubblicazione della serie Bard's Tale.

M.L. Mi rendo conto che si tratta di una domanda poco specifica, ma quanto o quale tipo di lavoro richiede la realizzazione di un programma come Bard's Tale? Che differenza intercorre fra questo ed un gioco che non deve fare parte di una serie o che non è un FRP?

Y. Il procedimento per realizzare un gioco come Bard's Tale II è diverso da quello necessario per creare un gioco singolo. Molti degli strumenti e delle caratteristiche ideate vengono continuamente riutilizzati.

F. Stabiliamo in anticipo rispetto ad altri programmi sia il codice, sia gli strumenti con cui lavorare, dato che sono assai simili a quelli della precedente versione. Produciamo anche dei programmi per facilitare la realizzazione della grafica, dei sotterranei (dungeon editor, n.d.r.) e più in generale di tutti quegli elementi che non servirebbero in un gioco singolo.

M.L. Che cos'è un dungeon editor?

F. Un dungeon editor per Bard's Tale è un sistema completo per la realizzazione di sotterranei e di labirinti, che permette di osservarli in una prospettiva tridimensionale, di stabilire dove esattamente vi siano

dei muri, delle porte e delle creature di cui si possono impostare anche nomi e caratteristiche: in pratica con esso si può controllare l'intero schema di gioco. Una volta ottenuto lo schema del labirinto prefissato si può trasferirlo direttamente nel programma principale. Il dungeon editor crea poi un codice utilizzabile in qualsiasi sistema. A questo punto non si rende necessario nessun adattamento di codice. Si tratta solo di una questione di gusto piuttosto che di tecnologia.

M.L. Cosa intendete per «qualsiasi sistema»?

Y. Per creare un gioco come Bard's Tale è necessaria una grande mole di lavoro; la Electronic Arts vende prodotti per molti computer diversi tra loro e comunque quasi sempre non compatibili. È logico perciò che se si dovesse scrivere un codice diverso per ogni computer sul quale si vuole far girare il proprio software, il lavoro diverrebbe immenso e la realizzazione di un gioco complesso come Bard's Tale risulterebbe impossibile. È per questo che ci si serve di strumenti che utilizzano moduli standard come il dungeon editor.

F. È il programmatore capo (Michael Cranford, n.d.r.) che realizza la maggior parte degli strumenti. Ha creato il dungeon editor ed il programma per gestire la grafica delle animazioni. Gli strumenti per creare il suono digitalizzato, invece, sono opera della EA e sono stati realizzati da Dave Warhol. La maggior parte di queste utility sono state sviluppate su di un sistema PC, nel quale carichiamo in memoria il codice relativo al sistema per il quale stiamo realizzando la versione del programma. Alcuni strumenti hanno una resa migliore proprio sul sistema per il quale stiamo creando il software, come ad esempio il suono digitalizzato.

Un altro programmatore si occupa esclusivamente dello sviluppo delle routine necessarie per il disco dati, mentre altri ancora sono specializzati nella produzione dei set di caratteri di cui ci serviamo.

M.L. Quando qualcuno produce per conto vostro parti di un programma, come è possibile conciliarle con i codici già sviluppati? Ad esempio, in che modo gestite la sezione sonora di un programma?

F. Per prima cosa determiniamo alcuni parametri tecnici, il programmatore sa quante voci ha a disposizione, quanto spazio in memoria e quanto su disco. Successivamente formuliamo i concetti fondamentali del gioco, in considerazione dei quali il programmatore deve svolgere il suo lavoro. Se l'ambientazione deve essere medioevale, indichiamo i titoli di alcuni film che ricreano scenari, ambienti ed un periodo storico simili a quelli presenti nel nostro gioco.

Y. Alcuni strumenti possono a volte essere utili anche per operazioni per le quali non erano stati originariamente concepiti. Alla EA ci siamo serviti del dungeon editor per creare mappe per eseguire test sul gioco. Abbiamo utilizzato le mappe dei labirinti per poter innescare un ciclo di prove di giocabilità che venivano eseguite in un tempo minore di quello richiesto da una normale partita. Ci sono così tante variabili relative agli imprevisti di una partita che, senza sapere dove essi si trovino e di quale evento si tratti, non si riuscirebbe mai a provarli tutti. Inoltre, non conoscendo in che modo si dovrebbe svolgere ogni singolo avvenimento, risulterebbe assai difficile scoprire la presenza di una bug. Nel labirinto vi sono alcune aree in cui non è possibile usare la magia e, se non le si conoscessero, si potrebbe pensare alla presenza di un errore nel programma. In pratica, insieme ad ogni nuova creazione, si sviluppa anche una precisa metodologia che consenta di effettuare dei test sul «Fantasy Role Playing (FRP, n.d.r.) stesso.

M.L. Sembra che, per questo tipo di prodotto, siano necessari molti test.

F. Se dovessi procedere a dei test su uno dei nostri prodotti, avrei bisogno di almeno quaranta ore per risolvere Bard's I e di ottanta per Bard's II. Se seguissi una diversa prassi non starei provando l'adventure correttamente. Se si imbocca subito il sentiero giusto, è ovvio che non si può provare per intero il corretto funzionamento del gioco.

Y. Ci servono più di duemila ore di test per provare Bard's Tale II e in verità, per un prodotto di questa complessità, ne sarebbero indispensabili il doppio o il triplo. Per comprimere i tempi, ci avvaliamo della nostra esperienza. I cicli dei test vengono suddivisi ed ognuno si specializza su una certa area. Ci sono molti elementi da controllare, come le utility, il trasferimento dei caratteri, i diversi aspetti dei labirinti, i finali dei giochi ed i loro bilanci, non facilitando nel contempo troppo la risoluzione e, naturalmente, verificando che tutte le caratteristiche del gioco operino correttamente. Per essere sicuri, ogni singola situazione viene controllata per alcune ore.

M.L. Con così tanti operatori coinvolti nello sviluppo com' è possibile organizzarsi in modo da sapere sempre chi, come e quando sta lavorando su di una data sezione?

F. Parte del mio lavoro consiste appunto nel coordinamento dei tempi e delle funzioni assegnate ad ognuno. Dal momento che il numero delle persone che lavorano ad uno stesso progetto è spesso elevato e poiché la maggior parte di queste sono programmatori indipendenti che, nello stesso tempo, prestano la loro opera per progetti diversi, devo per forza calcolare in precedenza i tempi da assegnare ad ogni gradino del progetto.

Y. Un elemento interessante che ho notato è che stiamo anche sviluppando dei procedimenti standard da seguire per realizzare questo tipo di prodotto. Ho scoperto che la produzione di Bard's Tale ha richiesto il

controllo di una serie di lavorazioni che dovevano rimanere entro linee di tempo tra loro convergenti. Per fare ciò ho sviluppato un sistema per registrare le revisioni ed i miglioramenti che sono stati eseguiti durante lo sviluppo.

Prima di poter iniziare la commercializzazione di un prodotto sono necessari molti altri elementi oltre al codice sul disco. La confezione, il manuale, le proiezioni sulle vendite, il posizionamento del programma all'interno di una strategia di mercato ben determinata, la pianificazione del budget pubblicitario e molte altre componenti rappresentano un momento altrettanto vitale quanto l'ideazione stessa di un nuovo programma.

I campi in cui operare sono dunque diversi e, sulla base dell'integrazione dei compiti all'interno del nostro team, non appena Brian mi comunica di aver conseguito un obiettivo prefissato posso smettere di occuparmene e passare al successivo.

Uno dei gradini più importanti all'interno del processo di produzione è sicuramente la «Alpha delivery». Giunti a quel punto, infatti, si è finalmente in grado di utilizzare il gioco con tutte le sue più importanti caratteristiche. Si può così iniziare a scrivere il manuale, a progettare la confezione e via dicendo. E, come nel caso di Brian, vi sono altri progetti per il cui sviluppo sono responsabile personalmente.

F. Mi piace paragonare il nostro lavoro a quello del mondo cinematografico. Abbiamo un produttore, un copione, un determinato budget, una star (il programmatore capo, n.d.r.) ed altri programmatori che contribuiscono alla realizzazione del progetto nel loro campo di specializzazione. Ci serviamo dei talenti ovunque li troviamo, ma c'è sempre una persona che riveste il ruolo di punto focale ed il programma è pur sempre la sua creazione alla quale fornisce vita ogni giorno di più.

M.L. Anche con tutta questa pianificazione e coordinazione sono sicuro che a volte si presenti qualche novità che possa mettere disordine nei vostri piani. Come riuscite ad ovviare a ciò?

F. In un lavoro creativo il prodotto viene continuamente ritoccato. Come può questo comparire in una pianificazione? Se troviamo qualche elemento che potrebbe veramente migliorare il programma, di sicuro facciamo il possibile per inserirlo.

Y. Non è possibile pianificare le innovazioni, a volte è necessario accettarlo come dato di fatto. Se si scopre che si tratta di una opportunità per migliorare il prodotto, non ho certo esitazioni nel bloccare tutto il lavoro per potervi inserire qualcosa di nuovo. Per questo la comunicazione tra i vari settori è così importante. La parte più delicata nello sviluppo di qualsiasi progetto è proprio la comunicazione fra tutti coloro che ne sono coinvolti. Penso che la maggior parte dei cambiamenti avvengano nel periodo successivo ad Alpha e precedente a Beta, quando cioè «blocchiamo» l'inserimento di novità in un prodotto per iniziare l'introduzione di piccoli miglioramenti e la ricerca di bug.

M.L. Quando avete deciso di fare di Bard's Tale una serie? Ed infine, come fate ad essere sicuri di creare un prodotto che garantisca di vendere con discreti risultati?

Y. Conosciamo piuttosto bene queste incognite. È un po' quanto accade con i fumetti. Una volta che il cliente ha stabilito che i prodotti sono di buon livello con l'acquisto, si è allora sicuri che la fiducia verrà accordata anche alle realizzazioni future. Abbiamo tenuto conto di ciò prima di iniziare Bard's II.

F. Una delle ragioni che stanno alla base di queste scelte consiste nel fatto che la gente si affeziona ai protagonisti ed ai caratteri del gioco e desidera quindi un seguito alle avventure dei suoi eroi che conduca all'episodio successivo della serie e a quello dopo ancora. Questa è una delle chiavi del successo del nostro settore.

Y. Questo ci riporta alla domanda iniziale, cioè quando abbiamo deciso di produrre più di un Bard's Tale. Data la grande quantità di lavoro e di energie spese nella realizzazione di ognuno di questi programmi abitualmente li sviluppiamo pensando sempre anche a delle altre versioni nelle quali potersi servire della tecnologia presente in quella precedente. Questo serve anche da stimolo e da giustificazione nell'uso di una maggiore quantità di tempo e nel provare nuove tecnologie per ottenere i risultati desiderati. Non sempre applichiamo questa metodologia di lavoro, ma un tipo di progetto di questo tipo porta inevitabilmente a pensare già dal primo momento ad una possibile continuazione dell'avventura.

Y. Il mio lavoro consiste nel fare «scommesse» intelligenti sui prodotti, cioè nell'analizzarli e nel cercare di capire se siano validi o meno. È l'istinto che mi guida ed è per questo che sono pagato. È comunque sempre presente un certo margine di rischio, dal momento che non possiamo certo prevedere con sicurezza il successo di vendita di un nuovo prodotto.

F. Mi capita di giocare molto spesso, sia con i miei prodotti, sia con quelli della concorrenza ed anche Joe fa altrettanto. Quando sono andato da lui dicendogli che avevo una buona idea per un prodotto, abbiamo scoperto che stavamo pensando alla stessa cosa.

Y. L'istinto deriva soprattutto dal fatto che ci divertiamo veramente con i nostri programmi e che uniamo a questo entusiasmo molta attenzione per ogni scelta. Sappiamo che, se avremo successo, centinaia di migliaia di persone potranno divertirsi coi nostri giochi e questa, oltre che una soddisfazione, è anche uno stimolo in più. Come ho però già accennato, molta della nostra cura e meticolosità deriva dal fatto che amiamo veramente questo tipo di prodotti. ■

GESTIONE DI UN CLUB

Utility per la registrazione e manipolazione di dati per C-64.

Di Enea-Spilimbergo Luca

Il mercato del software offre attualmente numerosi DATA BASE assai validi, ma frequentemente di difficile utilizzo da parte dell'utente medio. Spesso infatti, per utilizzare al meglio un qualsiasi programma, sviluppato da una delle tante software-house, che «sfornano» quasi quotidianamente utility e giochi per il C-64, è necessario leggere un consistente manuale, spesso e volentieri in lingua inglese. Capita poi che, dopo aver dedicato tempo prezioso allo studio delle caratteristiche del prodotto in esame, ci si accorga che esso non potrà mai soddisfare le nostre particolari esigenze.

Il programma che vi proponiamo non pretende certo di sostituire in toto il software in commercio, ma si prefigge lo scopo di fornire gli strumenti essenziali per l'organizzazione di un piccolo archivio, facilmente utilizzabile e modificabile (quindi adattabile alle proprie esigenze). Il listato riportato nelle pagine seguenti è relativo alle modalità di gestione di uno SKI CLUB, ma, come illustrato in seguito, sono sufficienti poche operazioni per adattarlo ad un altro modello organizzativo.

Passiamo all'analisi del funzionamento del programma. La prima schermata è relativa al menù principale: alla prima comparsa, ovvero senza possedere ancora dei dati archiviati, le uniche opzioni utilizzabili sono: INSERIMENTO RECORD, GESTIONE DISCO ed ovviamente FINE PROGRAMMA. Prendiamo quindi in esame le singole opzioni.

Inserimento dei record

Questa opzione permette di creare o estendere le dimensioni di un archivio. Ogni input, analogamente alle altre opzioni, viene controllato; in altri termini non vengono presi in considerazione tutti i tasti di controllo (cursori, colori, ecc.). Non è inoltre consentito non rispondere ad un input, vale a dire inserire una stringa nulla. Nel caso in cui il contenuto di un campo debba essere lasciato nullo sarà sufficiente premere la barra spaziatrice e «CR».

Ogni input è soggetto anche ad un controllo relativo alla sua lunghezza. Durante questa procedura sono disponibili i seguenti comandi: «↑» ritorna al campo precedente, consentendo correzioni, «F1» ritorna al menù cancellando il record attuale.

Esiste inoltre un controllo sul numero dei record inseriti: se questo supera il valore utilizzato per il dimensionamento, un messaggio opportuno invita l'utente a modificare la linea 260. Si consiglia peraltro di specificare un numero massimo di soci non troppo superiore a quello effettivo, questo per non rallentare eccessivamente l'esecuzione del programma. Infatti, predisponendo un array di dimensioni elevate, si costringe il sistema operativo ad eseguire lunghe ricerche degli spazi di cui necessita per le memorizzazioni temporanee.

Modifica dei record

Tramite questa procedura è possibile modificare e/o cancellare un record. Viene chiesto se si conosce il numero dei record da modificare; in caso negativo è possibile ricercarlo nell'archivio tramite la pressione dei tasti «+» e «-», che visualizzano rispettivamente il record successivo o quello precedente.

Premendo «M» si accede alla modifica, che permette di scegliere il campo da modificare per mezzo dei tasti cursore e «CR». A questo punto, mentre il programma attende l'immissione di nuovi dati, viene stampato ciò che precedentemente era contenuto nel campo stesso.

Il tasto «C», seguito da una conferma, cancella il record visualizzato e compatta il contenuto dell'archivio.

La pressione del tasto «R» permette l'accesso ad un altro record. Attraverso il tasto «F1» si può richiamare il menù principale.

Gestione del disco

In questo modo si accede ad un sottomenù che consente nu-

merose opzioni il cui ruolo è di facile comprensione. Veniamo dunque alla descrizione della più complessa, la funzione DIRECTORY, il cui uso permette di scegliere il tipo di directory che si vuole visualizzare. A questo proposito si rivela sicuramente opportuno fornire ragguagli più precisi. Il caricamento della directory può essere effettuato utilizzando la ricerca approssimata e sono quindi consentite directory del tipo:

$	directory completa
$*=P	solo file PRG
$*=U	solo file USR
$*=R	solo file REL
$PIP*	vedi manuale 1541 pag. 13
$P?P*	vedi manuale 1541 pag. 19
$P?P*=P	file di nome «P?P*» tipo PRG

Durante la visualizzazione della directory, esiste la possibilità di tornare al menù principale premendo il tasto funzione «F1». È inoltre possibile arrestare la visualizzazione della directory premendo un tasto qualsiasi (tranne ovviamente SHIFT, C=e CTRL), compiendo nuovamente la medesima operazione la visualizzazione continua.

Se la directory contiene più di 17 file vengono visualizzati i primi 17 e, dopo una breve pausa, i rimanenti. Per quanto riguarda la registrazione dei dati, è opportuno ricordare che sul disco ogni record è spezzato in due parti, in quanto non esiste la possibilità di scrivere più di 79 caratteri senza servirsi di appositi separatori, fatto imputabile a carenze progettuali del 1541 stesso.

Un'ultima indicazione: se durante la fase di registrazione il nome del file digitato è già presente nella directory, viene concessa la possibilità di aggiornarne (replace) o di modificarne il nome.

Ordinamento logico

Ordina l'archivio secondo uno dei campi; la scelta è effettuata tramite i tasti cursore e return. L'algoritmo utilizzato per l'ordinamento è il «bubble sort ottimizzato», che consente un veloce ordinamento nel caso in cui l'archivio sia già parzialmente ordinato.

Ricerca logica

Ricerca un record in cui il contenuto del campo scelto sia uguale al valore specificato. Indicando un numero di caratteri inferiore alla lunghezza consentita dal campo, vengono visualizzati tutti quei record in cui il campo scelto inizia per i caratteri inseriti. Una volta individuato il record è possibile modificarlo (tasto «M») o continuare la ricerca (tasto «C»).

Per quanto riguarda l'algoritmo utilizzato si è fatto uso di un metodo di ricerca a scansione diretta, evitando quello a scansione controllata (ricerca binaria o discontinua), per non complicare troppo la subroutine, dal momento che il tempo impiegato dal primo algoritmo per individuare un record, pur avendone in memoria un centinaio, è più che accettabile.

Stampa logica

Procedura analoga alla precedente, salvo il fatto che l'output è rivolto verso la stampante.

Stampa di etichette

Questa opzione stampa le etichette relative ai record presenti in memoria. La routine è in grado di individuare anche una famiglia stampando solo l'etichetta relativa al capofamiglia; per far ciò, però, è necessario che l'archivio sia stato precedentemente ordinato in due momenti distinti, prima secondo il campo COGNOME, quindi secondo il campo TELEFONO (questo facendo sempre riferimento ai campi utilizzati come esempi nel listato).

Come adattare il programma alle proprie esigenze

Il procedimento è molto semplice: il nome del CLUB va inserito alla linea 240. Poi si modifica, se necessario, il numero dei campi per record e quello massimo dei soci. È quindi sufficiente mutare il nome dei campi alle righe 370-380, la loro lunghezza (riga 350) e le coordinate alle quali posizionare il cursore per l'input (310-330). Se qualche campo ha contenuti «obbligatori» (esempio: SI o NO) è sufficiente controllarne il contenuto alle righe 10145 e 11390, specificando il numero del campo.

Un ultimo particolare di rilievo riguarda il fatto che nella routine di stampa, alle righe 42550-42610, vengono utilizzate delle abbreviazioni per i nomi dei campi da inserire nella testata, che possono venire tranquillamente modificate in relazione ai nomi dei campi usati.

Tipo di stampante utilizzata

Le routine di stampa sono dedicate alla RITEMAN C+, ma non è difficile renderle compatibili con quelle necessarie al funzionamento di stampanti diverse. È necessario tenere presente che la RITEMAN viene settata dal programma a 132 colonne, questo per consentire la stampa di un record su un'unica riga.

I caratteri di controllo inviati alla stampante hanno i seguenti significati:

ESC+CHR $(88) +CHR$ (1)	attiva la stampa in NLQ
ESC+CHR $(88) +CHR$ (0)	disattiva la stampa in NLQ
CHR$ (15)	seleziona la stampa a 132 colonne
ESC + «G»	seleziona la stampa in DOUBLE STRIKE (grassetto)
CHR$ (14)	seleziona la stampa in 40 colonne, se è già attivo il modo 132 cpl stampa 66 cpl
ESC+«D»+CHR$ (0)	setta la tabulazione orizzontale
CHR$ (9)	esegue la tabulazione orizzontale.

Il programma contiene inoltre

alcune subroutine facilmente utilizzabili nei propri programmi, a questo scopo nella Tavola 1 è presente l'elenco delle subroutine completo di descrizione della funzione implementata, linee di programma occupate, parametri di passaggio e variabili modificate. Per problemi o modifiche è possibile contattare direttamente l'autore: ENEA-SPILIMBERGO Luca, Via Pablo Neruda 4/A - 20060 Bussero (Mi).

Il listato è disponibile già digitato su disco. Per ordini telefonare ai seguenti numeri: 02/794181-799492.

TAVOLA 1: ELENCO SUBROUTINE

NOME SUBROUTINE	LINEE DI PROGR. INIZIO	FINE	DESCRIZIONE FUNZIONE	INPUT	OUTPUT	VARIABILI MODIFICATE
STAMPA «MENÙ»	35050	30590	Stampa la scritta «MENÙ» utilizzando i caratteri grafici.	nessuno	nessuno	nessuna
PRINT AT	31000	31070	Posiziona il cursore alle coordinate X e Y, riga e colonna.	X,Y	nessuno	nessuna
GESTIONE OPZIONI	40000	40220	Stampa No opzioni alle coordinate contenute nel loro nome. Aspetta scelta dell'utente tramite una barra in reverse.	NO,OP$	SC	I,S$
CONTROLLO TASTO PREMUTO	45000	45120	Aspetta la pressione di uno dei tasti contenuti in CA$ e ritorna la sua posizione nella stringa.	CA$	CH	L0,X1$,X9
INPUT CONTROLLATO	46000	46110	Accetta LM caratteri eccetto: [DS], [SIN], [SÙ], [GIÙ], [CR], [CLR], [HOME] premendo [F1] si torna al MENÙ.	LM	I$	X$,LM
CREA FINESTRA VIDEO	52000	52080	Stampa una finestra vidoo 30x8 alle coordinate X, Y.	X,Y	nessuno	nessuna
STAMPA MASCHERA PER INPUT O MODIFICA RECORD	59110	59100	Stampa tutti i nomi dei campi e divide lo schermo in due finestre.	NC, X1 ( )	nessuno	I,X, Y
CONTROLLO ERRORE DISCO	59500	59600	Legge l'errore del canale 15. Se presente, stampa una finestra contenente il tipo di errore.	nessuno	nessuno	NE, ME$, TE, SE, P, X, Y, XT, YT
PULISCE SCHERMO	60000	60070	Pulisce lo schermo senza alterare la finestra video. Stampa anche il nome dell'opzione corrente (ST$).	ST$	nessuno	Y1,X,Y
ASPETTA LA PRESSIONE DI UN TASTO	61000	61070	Invita l'utente a premere un tasto facendo lampeggiare la scritta PREMI UN TASTO.	XT,YT	nessuno	X,Y,A$,R$
DIRECTORY	24500	24645	Visualizza la directory. Consente la scelta del tipo di directory (es. solo PRG, SEQ, ecc.).	nessuno	nessuno	X,Y,LM,D$ N1,B$,C$,T$, XT,YT,C

Listato. *Gestione Club*

```
10 REM ----------------------------
20 REM ------     GESTIONE
   ------
30 REM -------------------------
   --------
40 REM -------      DI   UN
   --------
50 REM -------
60 REM -------       CLUB
70 REM -------
80 REM -------        DI
90 REM ------------------------
   -------
100 REM------- ENEA-SPILIMBERGO
    -------
110 REM--------------------------
120 REM--------    LUCA
130 REM--------------------------
140 REM--------  VIA NERUDA 4/A
    -------
150 REM--------------------------
160 REM-------   20060 BUSSERO
    ------
170 REM--------------------------
180 REM-------   TEL.02/95038470
    -------
190 REM--------------------------
200 REM--------------------------
210 :
220 REM  ** ASSEGNAZIONE VARIAB
    ILI **
230 :
240 NS$=" SKI CLUB BUSSERO ":RE
    M NOME CLUB
250 NC=9:REM NUMERO CAMPI PER R
    ECORD
260 NP=150:REM NUMERO MAX PERSO
    NE
290 :
300 :
310 DATA 4,5,6,7,9,11,15,16,18
320 REM ↑↑↑ COORDINATE X PER IN
    PUT ↑↑↑
330 DATA 13,13,13,13,13,18,25,2
    5,25
340 REM ↑↑↑ COORDINATE Y PER IN
    PUT ↑↑↑
350 DATA 20,15,25,20,11,8,7,5,1
360 REM ↑↑↑ LUNGHEZZA CAMPI
        ↑↑↑
370 DATA COGNOME,NOME,INDIRIZZO
    ,LOCALITA',TELEFONO,DATA DI
     NASCITA
375 DATA "N°TESSERA F.I.S.I."
380 DATA "N°TESSERA SKI CLUB",V
    ECCHIO SOCIO      [S/N]
385 REM ↑↑↑ NOMI CAMPI
        ↑↑↑
390 :
400 REM ** DIMENSIONAMENTO ARRA
    Y **
410 :
420 DIM RE$(NP),OP$(NC),X1(NC),
    Y1(NC),L1(NC),IN(NC+1),G(NP
    )
430 :
440 REM ** LETTURA DATI ARRAY *
    *
445 :
460 FOR I=1 TO NC:READ X1(I):NE
    XT I
470 FOR I=1 TO NC:READ Y1(I):NE
    XT I
480 FOR I=1 TO NC:READ L1(I):NE
    XT I
485 IN=1:IN(1)=IN:FOR I=2 TO NC
    :IN=IN+L1(I-1):IN(I)=IN:NEX
    T I
486 IN(NC+1)=IN(NC)+L1(NC)
490 FOR I=1 TO NC:READ CA$(I):N
    EXT I
492 OPEN 15,8,15:REM CANALE ERR
    ORE
495 POKE53280,0:POKE53281,0:PRI
    NT"■■";
497 GOTO 520
499 :
500 REM ** MENU' PRINCIPALE **
505 :
510 GOSUB 60000
520 ST$=" MENU' PRINC.        "
525 PRINT"■■";:GOSUB 30000:GOSUB
     30500
530 NO=8:REM NUMERO OPZIONI MEN
    U
540 OP$(1)="1007INSERIMENTO REC
    ORD."
550 OP$(2)="1009MODIFICA RECORD
    ...."
560 OP$(3)="1011GESTIONE DISCO.
    ...."
570 OP$(4)="1013ORDINAMENTO LOG
    ICO."
580 OP$(5)="1015RICERCA LOGICA.
    ...."
590 OP$(6)="1017STAMPA LOGICA..
    ...."
600 OP$(7)="1019STAMPA ETICHETT
    E."
610 OP$(8)="1021FINE PROGRAMMA.
    .."
620 ME=(1=1):GOSUB 40000
630 :
700 REM ** RICHIAMO ROUTINE SCE
    LTA **
710 :
720 ON SC GOTO 10000,11000,2000
    0,41000,43000,42000,26000
730 CLOSE 15
740 PRINT"■FINE GESTIONE CLUB■■
    ■"
750 END
10000 REM ====================
      ========
10010 REM ===  INSERIMENTO RECO
      RDS ===
10020 REM ====================
      ========
10030 :
10032 IF NR<NP THEN 10045
10033 X=06:Y=05:GOSUB 52000
10034 X=07:Y=07:GOSUB 31000:PRI
      NT"ARCHIVIO COMPLETO!!"
10035 X=09:Y=07:GOSUB 31000:PRI
      NT"PER MEMORIZZARE UN NUM
      ERO"
10036 X=10:Y=07:GOSUB 31000:PRI
      NT"MAGGIORE DI RECORDS BI
      SOGNA"
10037 X=11:Y=07:GOSUB 31000:PRI
      NT"MODIFICARE IL VALORE D
      I NP"
10038 X=12:Y=07:GOSUB 31000:PRI
      NT"ALLA LINEA 260 . "
10040 XT=14:YT=13:GOSUB 61000
10042 GOTO 500
10045 ST$=" INSERIMENTO RECORDS
       "
10050 GOSUB 60000:NR=NR+1:RE$(N
      R)=""
10052 N$=MID$(STR$(NR),2):N$=LE
      FT$("   ",3-LEN(N$))+N$
10055 X=01:Y=32:GOSUB 31000:PRI
      NT"■REC. ";N$;"■";
10060 X=01:Y=01:GOSUB 31000:PRI
      NT"■          ■ ■ INSERIMENT
      O RECORDS "
10070 X=21:Y=04:GOSUB 31000:PRI
      NT" ■PREMI 'F1' PER TORNA
      RE AL MENU'■ "
10080 GOSUB 59000:REM STAMPA MA
      SCHERA
10090 REM INPUT DATI
10100 FOR I=1 TO NC
10110 X=X1(I):Y=Y1(I):GOSUB 310
      00:PRINT"";
10120 LM=L1(I):GOSUB 46000
10125 IF I$="↑" AND I>1 THEN RE
      $(NR)=LEFT$(RE$(NR),IN(I-
      1)-1):I=I-1:GOTO10110
10130 IF I$="↑"THEN 10110
10140 GOSUB 62000
10145 IF I=9 AND I$<>"N" AND I$
      <>"S" THEN 10110
10150 RE$(NR)=RE$(NR)+I$:G(NR)=
      NR
10160 NEXT I
10170 FOR DE=1 TO 300:NEXT DE
10180 GOTO 10000
10190 :
11000 REM ====================
      ========
11010 REM ===     MODIFICA RECOR
      D
11020 REM ====================
      ========
11030 :
11035 IF G(1)=0 AND NR<>0 THEN
      FOR I=1 TO NR:G(I)=I:NEXT
      I
11040 ST$=" MODIFICA RECORDS
       "
11050 GOSUB 50000:GOSUB 60000
11060 X=12:Y=02:GOSUB 31000:PRI
      NT"CONOSCI IL NUMERO DEL
      RECORD [S/N]";
11070 CA$="SN":GOSUB 45000
11080 IF CH=2 THEN 11130
11090 X=12:Y=02:GOSUB 31000:PRI
      NT" INSERISCI IL NUMERO D
      EL RECORD  ";
11100 LM=3:GOSUB 46000:R=INT(VA
      L(I$))
11110 IF R<=0 OR R>NR THEN 1109
      0
11120 GOTO 11140
11130 R=1
11140 GOSUB 60000
11150 :
11160 X=01:Y=01:GOSUB 31000:PRI
      NT"■          ■ ■     MODIFICA
       RECORDS      ■"
11170 GOSUB 59000:REM STAMPA MA
      SCHERA
11175 N$=MID$(STR$(R),2):N$=LEF
      T$("   ",3-LEN(N$))+N$
11177 X=01:Y=32:GOSUB 31000:PRI
      NT"■REC. ";N$;"■";
11180 FOR J=1 TO NC
11190 X=X1(J):Y=Y1(J):GOSUB 310
      00:PRINTMID$(RE$(G(R)),IN
      (J),L1(J));
11200 NEXT J
11210 X=22:Y=04:GOSUB 31000:PRI
      NT"—— [M],[C],[+],[-],[F1
      ],[R] ——";
11220 CA$="R■M+C-":GOSUB 45000
11230 ON CH GOTO 11265,11260,11
```

```
       270,11250,11500
11240 IF R>1  THEN R=R-1:GOTO 1
       1175
11245 GOTO 11220
11250 IF R<NR THEN R=R+1:GOTO 1
       1175
11255 GOTO 11220
11260 GOTO 500
11262 REM ACCESSO AL RECORD N'.
       ..
11263 :
11265 GOSUB 60000:GOTO 11090
11270 REM *** MODIFICA ***
11275 X=22:Y=03:GOSUB 31000:PRI
       NT"—— SCEGLI IL CAMPO PR
       EMI <CR> ——"
11280 FOR I=1 TO NC
11290 PX$=MID$(STR$(X1(I)),2):I
       F LEN(PX$)<2THENPX$="0"+P
       X$
11300 OP$(I)="02"+PX$+CA$(I)
11310 NEXT:NO=NC:GOSUB 40080
11320 Y=05:X=06:GOSUB 52000
11330 X=07:Y=06:GOSUB 31000:PRI
       NT"CAMPO ◘";CA$(SC);"■";
11340 X=09:Y=06:GOSUB 31000:PRI
       NT"VECCHIO CONTENUTO:";
11350 X=10:Y=06:GOSUB 31000:PRI
       NTMID$(RE$(G(R)),IN(SC),L
       1(SC));
11360 X=12:Y=06:GOSUB 31000:PRI
       NT"NUOVO CONTENUTO:"
11370 X=13:Y=06:GOSUB 31000:PRI
       NT"";:LM=L1(SC):GOSUB 460
       00
11380 GOSUB 62000
11390 IF SC=9 AND I$<>"N" AND I
       $<>"S" THEN 11370
11400 RE$(G(R))=LEFT$(RE$(G(R))
       ,IN(SC)-1)+I$+MID$(RE$(G(
       R)),IN(SC+1))
11410 GOTO 11140
11420 :
11500 REM **** CANCELLA RECORD
       R ****
11510 :
11520 X=07:Y=07:GOSUB 52000
11530 X=10:Y=09:GOSUB 31000:PRI
       NT"CANCELLO ? [S/N] ";
11540 CA$="SN":GOSUB 45000:PRIN
       T"◘";MID$(CA$,CH,1);"■"
11550 ON (3-CH) GOTO 11140
11560 REM OK! CANCELLO...
11570 RE$(G(R))=RE$(NR)
11580 FOR I=1 TO NR
11590 ::IF G(I)=G(NR) THEN G(I)
       =G(R):GOTO 11610
11600 ::IF G(I)=G(R) THEN P1=I
11610 NEXT I
11615 IF R=NR THEN P1=NR
11620 NR=NR-1:R=R-1
11630 FOR I=P1 TO NR
11640 ::G(I)=G(I+1)
11650 NEXT I
11660 GOTO 11140
11670 :
20000 REM ====================
       ========
20010 REM ===     GESTIONE DISCO
        ===
20020 REM ====================
       ========
20030 :
20040 ST$=" GESTIONE DISCO"
20045 GOSUB 60000
20050 X=02:Y=11:GOSUB 31000:PRI
       NT"◘ GESTIONE DISCO ■"
20055 NO=8
20060 OP$(1)="1007LETTURA DATI.
       ........"
20070 OP$(2)="1009REGISTRAZIONE
        DATI."
20080 OP$(3)="1011CANCELLAZIONE
        FILES."
20090 OP$(4)="1013MODIFICA NOME
        FILES."
20100 OP$(5)="1015VALIDATE DEL
       DISCO..."
20105 OP$(6)="1017DIRECTORY....
       "
20110 OP$(7)="1019ALTRO COMANDO
        DOS."
20112 OP$(8)="1021TORNA AL MENU
       ' "
20115 GOSUB 40000
20120 ON SC GOTO 21000,22000,23
       000,24000,25000,24500,255
       00,500
21000 REM ====================
       ========
21010 REM ===      LETTURA DATI
         ===
21020 REM ====================
       ========
21030 :
21035 NR=0:REM CANCELLA FILE IN
        MEMORIA
21040 GOSUB 60000
21050 X=02:Y=13:GOSUB 31000:PRI
       NT"◘ LETTURA DATI ■"
21060 X=06:Y=02:GOSUB 31000:PRI
       NT"NOME DEL FILE.....";
21070 LM=16:GOSUB 46000:NF$=I$
21080 OPEN 1,8,5,"0:"+NF$+",S,R
       "
21090 X=08:Y=04:GOSUB 31000:PRI
       NT"ATTENDI PREGO......";
21100 GOSUB 59500
21110 NR=NR+1:G(NR)=NR
21120 INPUT#1,R1$,R2$
21130 RE$(NR)=LEFT$(R1$,LEN(R1$
       )-1)+RIGHT$(R2$,LEN(R2$)-
       1)
21140 IF ST=0 THEN 21110
21150 CLOSE 1
21160 X=08:Y=03:GOSUB 31000:PRI
       NT"OK OPERAZIONE CONCLUSA
       "
21170 XT=12:YT=10:GOSUB 61000
21180 GOTO 20000
22000 REM ====================
       ========
22010 REM ===     REGISTRAZIONE D
       ATI ===
22020 REM ====================
       ========
22030 :
22040 GOSUB 50000
22050 GOSUB 60000:CLOSE1
22060 X=02:Y=09:GOSUB 31000:PRI
       NT"◘ REGISTRAZIONE DATI ■
       "
22070 X=06:Y=02:GOSUB 31000:PRI
       NT"NOME DEL FILE.....";
22075 LM=16:GOSUB 46000:NF$=I$:
       M$="0:"
22080 OPEN 1,8,5,M$+NF$+",S,W"
22085 X=10:Y=08:GOSUB 31000:PRI
       NT"ATTENDI PREGO......";
22090 GOSUB 59500
22095 IF NE=0 THEN 22120
22100 X=13:Y=01:GOSUB 31000:PRI
       NT"FILE ESISTENTE. REPLAC
       E ? [S/N] ";
22113 CA$="SN":GOSUB 45000:ON (
       3-CH) GOTO 22050
22115 M$="@0:":GOTO 22080
22120 IF G(1)=0 AND NR<>0 THEN
       FOR I=1 TO NR:G(I)=I:NEXT
        I
22125 :
22130 FOR I=1 TO NR
22140 ::R1$=LEFT$(RE$(G(I)),IN(
       5)-1)+"*"
22150 ::R2$="*"+MID$(RE$(G(I)),
       IN(5))
22160 ::PRINT#1,R1$;CHR$(13);R2
       $
22170 NEXT I
22180 CLOSE 1
22190 X=10:Y=03:GOSUB 31000:PRI
       NT"OK OPERAZIONE CONCLUSA
       ";
22200 XT=17:YT=10:GOSUB 61000
22210 GOTO 20000
22220 :
23000 REM ====================
       ========
23010 REM ===  CANCELLAZIONE FI
       LES ===
23020 REM ====================
       ========
23030 :
23040 GOSUB 60000
23050 X=02:Y=09:GOSUB 31000:PRI
       NT"◘ CANCELLAZIONE FILES
       ■"
23060 X=06:Y=02:GOSUB 31000:PRI
       NT"NOME DEL FILE.....";
23070 LM=16:GOSUB 46000:NF$=I$
23080 X=08:Y=02:GOSUB 31000:PRI
       NT"SEI SICURO [S/N]..";
23090 CA$="SN":GOSUB 45000:ON (
       3-CH) GOTO 23060:PRINT" ◘
       ";MID$(CA$,CH,1);"■";
23100 PRINT#15,"S0:"+NF$
23110 GOSUB 59500
23120 XT=17:YT=11:GOSUB 61000
23130 GOTO 20000
23140 :
24000 REM ====================
       ========
24010 REM ===  MODIFICA NOME FI
       LE   ===
24020 REM ====================
       ========
24030 :
24040 GOSUB 60000
24050 X=02:Y=09:GOSUB 31000:PRI
       NT"◘ MODIFICA NOME FILE ■
       "
24060 X=06:Y=02:GOSUB 31000:PRI
       NT"VECCHIO NOME....";
24070 LM=16:GOSUB 46000:VN$=I$
24080 X=08:Y=02:GOSUB 31000:PRI
       NT"NUOVO NOME......";
24090 LM=16:GOSUB 46000:NN$=I$
24100 CO$="R0:"+NN$+"=0:"+VN$
24105 PRINT#15,CO$
24110 GOSUB 59500:REM CONTROLLA
        ERRORE
24115 X=12:Y=07:GOSUB 31000:PRI
       NT"OK! RENAME CONCLUSO"
24120 XT=17:YT=11:GOSUB 61000
24130 GOTO 20000
24140 :
24500 REM ====================
       ========
24510 REM ====       DIRECTORY
       ====
24520 REM ====================
       ========
24525 :
24530 GOSUB 60000
24531 X=02:Y=13:GOSUB 31000:PRI
       NT"◘ DIRECTORY ■"
24532 X=04:Y=02:GOSUB31000:PRIN
       T"POSSIBILI DIRECTORY:"
24535 X=06:GOSUB31000:PRINT"$0
        $*=P (S,U,R)  $NOME*  $N
       ?M*"
24536 LM=18:X=08:GOSUB31000:PRI
       NT"===> ";:GOSUB46000:D$=
       I$
24538 GOSUB 60000
24540 OPEN 1,8,0,D$
24542 N1=0
```

```
24545 GET#1,A$,B$
24550 GET#1,A$,B$
24555 GET#1,A$,B$
24560 C=0
24565 IF A$<>"" THEN C=ASC(A$)
24570 IF B$<>"" THEN C=C+ASC(B$
      )
24571 N1=N1+1
24573 IF N1>17 THEN N1=0:FORI=1
      TO3000:NEXT:GOSUB 60000
24575 X=1+N1:Y=7:GOSUB31000:PRI
      NTMID$(STR$(C),2)
24577 Y=12:GOSUB31000:PRINT"";
24580 GET#1,B$:IFST<>0 THEN 246
      25
24585 IF B$<>CHR$(34) THEN 2458
      0
24590 GET#1,B$:IF B$<>CHR$(34)
      THEN PRINTB$;:GOTO 24590
24595 GET#1,B$:IF B$=CHR$(32) T
      HEN 24595
24600 Y=30:GOSUB31000:PRINT"";:
      C$=""
24605 C$=C$+B$:GET#1,B$:IF B$<>
      "" THEN 24605
24610 PRINTLEFT$(C$,3)
24612 GET T$
24615 IF T$="■" THEN CLOSE1:GOT
      O 500:REM AL MENU'
24617 IF T$<>"" THEN POKE198,0:
      WAIT198,1:POKE198,0
24620 IF ST=0 THEN 24550
24625 PRINT"BLOCKS FREE."
24630 CLOSE1
24635 XT=21:YT=11:GOSUB 61000
24640 GOTO 20000
24645 :
25000 REM ==================
      ========
25010 REM ===  VALIDATE DEL DIS
      CO ===
25020 REM ==================
      ========
25030 :
25040 GOSUB 60000
25050 X=02:Y=09:GOSUB 31000:PRI
      NT"■ VALIDATE DEL DISCO ■
      "
25060 X=05:Y=03:GOSUB 31000:PRI
      NT"L'OPERAZIONE DI VALIDA
      TE SI RENDE"
25070 X=06:Y=03:GOSUB 31000:PRI
      NT"NECESSARIA DOPO UN FRE
      QUENTE USO"
25080 X=07:Y=03:GOSUB 31000:PRI
      NT"DEI COMANDI SCRATCH E
      RENAME.
25090 X=08:Y=03:GOSUB 31000:PRI
      NT"DOVRAI ATTENDERE CIRCA
       2 MINUTI..."
25100 PRINT#15,"V"
25110 CLOSE 15:OPEN 15,8,15
25120 X=12:Y=11:GOSUB 31000:PRI
      NT"OK! VALIDATE TERMINATO
      "
25130 XT=17:YT=11:GOSUB 61000
25140 GOTO 20000
25150 :
25500 REM ==================
      ========
25510 REM ====  ALTRO COMANO DO
      S ===
25520 REM ==================
      ========
25530 :
25540 GOSUB 60000
25550 X=02:Y=09:GOSUB 31000:PRI
      NT"■ COMANDO DOS DIRETTO
      ■"
25560 X=06:Y=02:GOSUB 31000:PRI
      NT"COMANDO....";
25570 LM=16:GOSUB 46000:CO$=I$
25580 PRINT#15,CO$
25590 GOSUB 59500:REM CHECK DIS
      K ERROR
25600 XT=17:YT=11:GOSUB 61000
25610 GOTO 20000
25620 :
26000 REM ==================
      ========
26010 REM ===    STAMPA ETICHETT
      E ===
26020 REM ==================
      ========
26030 :
26040 GOSUB 50000
26050 GOSUB 60000
26060 X=02:Y=10:GOSUB 31000:PRI
      NT"■ STAMPA ETICHETTE ■"
26070 X=10:Y=10:GOSUB 31000:PRI
      NT"ATTENDI PREGO...."
26080 :
26090 OPEN 1,4
26100 PRINT#1,CHR$(27);CHR$(88)
      CHR$(1);:REM NLQ
26110 IF ST<>0 THEN CLOSE1:GOSU
      B 40500:GOTO 500
26120 FOR J=1 TO NR
26130 ::CO$=MID$(RE$(G(J)),IN(1
      ),L1(1))
26140 ::IO$=MID$(RE$(G(J)),IN(3
      ),L1(3))
26145 ::TO$=MID$(RE$(G(J)),IN(5
      ),L1(5))
26150 REM REPEAT
26160 ::::I=J+1
26170 ::::C1$=MID$(RE$(G(I)),IN
      (1),L1(1))
26180 ::::I1$=MID$(RE$(G(I)),IN
      (3),L1(3))
26185 ::::T1$=MID$(RE$(G(I)),IN
      (5),L1(5))
26190 ::::IF ((CO$=C1$ OR TO$=T
      1$)AND IO$=I1$) THEN I=I+
      1:GOTO 26170
26195 REM UNTIL (CO$=C1$ AND IO
      $=I1$)
26200 REM FINE (COGNOMI O N' TE
      LEFONICI) E INDIRIZZI UGU
      ALI
26210 ::IF I=J+1 THEN PR=J:GOSU
      B 27000:GOTO 26310:REM SO
      LO UNA PERSONA
26220 REM PERSONA TROVATA - CER
      CA CAPOFAMIGLIA
26230 :MA$="99.99.99":PM=0
26240 ::FOR K=J TO I-1
26250 ::::E1$=MID$(RE$(G(K)),IN
      (6),L1(6))
26260 ::::E1$=RIGHT$(E1$,2)+MID
      $(E1$,3,4)+LEFT$(E1$,2)
26270 ::::IF E1$<MA$ THEN MA$=E
      1$:PM=K
26280 ::NEXT K
26285 ::J=I-1
26290 ::PR=PM
26300 ::GOSUB 27000
26310 NEXT J
26320 PRINT#1:CLOSE1
26330 XT=16:YT=14:GOSUB 61000
26340 GOTO 500
26350 :
27000 REM STAMPA ETICHETTA PER
      BUSTA POSTALE
27010 :
27020 C$=MID$(RE$(G(PR)),IN(1),
      L1(1)):P1=LEN(C$)
27025 IF MID$(C$,P1,1)=" "THENP
      1=P1-1:GOTO27025
27027 NC$=LEFT$(C$,P1+1)+MID$(R
      E$(G(PR)),IN(2),L1(2))
27030 IN$=MID$(RE$(G(PR)),IN(3)
      ,L1(3))
27040 LO$=MID$(RE$(G(PR)),IN(4)
      ,L1(4))
27050 :
27060 PRINT#1,"     ";NC$;CHR$(1
      3)
27070 PRINT#1,"     ";IN$
27080 PRINT#1,"     ";LO$
27090 FOR LF=1 TO 5:PRINT#1:NEX
      T
27100 RETURN
27110 :
30000 REM ** SUB DISEGNA BORDO
      **
30010 PRINT"╭";
30020 GOSUB 30200
30030 PRINT"╮";
30040 FOR I=1 TO 21
30050 ::PRINT"|"SPC(38)"|";
30060 NEXT I
30070 PRINT"╰";
30080 GOSUB 30200
30090 PRINT"╯";
30100 PRINT
30110 PRINT"STATO  ===>";ST$;"■
      ■";
30120 RETURN
30200 FOR I=1 TO 38
30210 ::PRINT"─";
30220 NEXT I
30230 RETURN
30240 :
30250 :
30500 REM ** STAMPA "MENU'" **
30510 :
30520 X=02:Y=08:GOSUB 31000
30530 PRINT"  ▄▄■▀
      ▀■▄▄"
30540 X=03:Y=07:GOSUB 31000
30550 PRINT"■        M E N U'
      ■"
30560 X=04:Y=08:GOSUB 31000
30570 PRINT"  ▀▀■▄
      ▄■▀▀"
30580 RETURN
30590 :
31000 REM ** PRINT AT **
31010 :
31020 POKE 783,PEEK(783) AND 25
      4
31030 POKE 781,X
31040 POKE 782,Y
31050 SYS 65520
31060 RETURN
31070 :
40000 REM ** GESTIONE OPZIONI *
      *
40010 :
40030 FOR I=1 TO NO
40040 ::Y=VAL(LEFT$(OP$(I),2))
40050 ::X=VAL(MID$(OP$(I),3,2))
40060 ::GOSUB 31000:PRINT MID$(
      OP$(I),5)
40070 NEXT I
40080 SC=1
40090 Y=VAL(LEFT$(OP$(SC),2))
40100 X=VAL(MID$(OP$(SC),3,2))
40110 GOSUB 31000:PRINT"■";MID$
      (OP$(SC),5);"■"
40120 POKE198,0:WAIT198,1:GET S
      $
40125 IF S$="■" AND NOT(ME)THEN
       500
40130 IF S$<>"▔" AND S$<>"■" AN
      D S$<>CHR$(13) THEN 40120
40140 GOSUB 31000:PRINT MID$(OP
      $(SC),5)
40150 IF S$=CHR$(13) THEN ME=(1
      =2):RETURN
40160 IF S$="▔" AND SC>1 THEN S
      C=SC-1:GOTO40090
```

```
40170 IF S$="]" AND SC>1 THEN S
      C=SC-1:GOTO 40090
40180 IF S$="]" AND SC=1 THEN S
      C=NO:GOTO 40090
40190 IF S$="▮" AND SC<NO THEN
      SC=SC+1:GOTO 40090
40200 IF S$="▮" AND SC=NO THEN
      SC=1
40210 GOTO40090
40220 :
40500 REM ** MESSAGGIO PER STAM
      PANTE **
40505 REM **      NON COLLEGATA
        **
40510 :
40520 X=07:Y=05:GOSUB 52000
40530 X=09:Y=08:GOSUB 31000:PRI
      NT"STAMPANTE NON COLLEGAT
      A!"
40540 X=10:Y=08:GOSUB 31000:PRI
      NT"ACCENDILA O CONTROLLA
      LE"
40550 X=11:Y=08:GOSUB 31000:PRI
      NT"CONNESSIONI......"
40560 XT=13:YT=12:GOSUB 61000
40570 RETURN
40580 :
41000 REM ===================
      ========
41010 REM ==   ORDINAMENTO LOGI
      CO   ==
41020 REM ===================
      ========
41030 :
41040 ST$=" ORDINAMENTO LOGICO
      "
41050 GOSUB 50000:GOSUB 60000
41060 GOSUB 59000
41070 X=02:Y=02:GOSUB 31000:PRI
      NT"SCEGLI IL CAMPO CHIAVE
       E PREMI <CR>"
41080 FOR I=1 TO NC
41090 PX$=MID$(STR$(X1(I)),2):I
      F LEN(PX$)<2THENPX$="0"+P
      X$
41100 OP$(I)="02"+PX$+CA$(I)
41110 NEXT:NO=NC:GOSUB 40080
41120 :
41125 X=05:Y=06:GOSUB 52000
41130 X=08:Y=11:GOSUB 31000:PRI
      NT"ORDINAMENTO IN CORSO"
41135 X=12:Y=13:GOSUB 31000:PRI
      NT"ATTENDERE PREGO....."
41140 REM ** SORT **
41150 REM SC=N' CAMPO CHIAVE
41155 :
41160 FOR X0=NR-1 TO 2 STEP -1
41165 ::FL=0
41170 ::FOR X1=1 TO X0
41180 ::::R1$=MID$(RE$(G(X1)),I
      N(SC),L1(SC))
41190 ::::R2$=MID$(RE$(G(X1+1))
      ,IN(SC),L1(SC))
41195 ::::IF SC=6 THEN R1$=RIGH
      T$(R1$,2)+MID$(R1$,4,2)+L
      EFT$(R1$,2)
41196 ::::IF SC=6 THEN R2$=RIGH
      T$(R2$,2)+MID$(R2$,4,2)+L
      EFT$(R2$,2)
41200 ::::IF R1$<=R2$ THEN 4122
      0
41210 ::::G0=G(X1):G(X1)=G(X1+1
      ):G(X1+1)=G0:FL=1
41220 ::NEXT X1
41230 ::IF FL=0 THEN X0=1
41240 NEXT X0
41250 XT=13:YT=14:GOSUB 61000
41260 GOTO 500
42000 REM ===================
      ========
42010 REM ==      STAMPA LOGICA
        ==
```

```
42020 REM ===================
      ========
42030 RI=(2=1):T$="STAMPA"
42040 ST$=" STAMPA LOGICA
      "
42050 GOSUB 50000:GOSUB 60000
42060 GOSUB 59000
42070 X=02:Y=02:GOSUB 31000:PRI
      NT"SCEGLI IL CAMPO CHIAVE
       E PREMI <CR>"
42080 FOR I=1 TO NC
42090 PX$=MID$(STR$(X1(I)),2):I
      F LEN(PX$)<2THENPX$="0"+P
      X$
42100 OP$(I)="02"+PX$+CA$(I)
42110 NEXT:NO=NC:GOSUB 40080
42120 :
42125 X=05:Y=06:GOSUB 52000
42130 X=07:Y=07:GOSUB 31000:PRI
      NT"CAMPO CHIAVE:"
42140 X=08:Y=07:GOSUB 31000:PRI
      NT"▮";CA$(SC);"▮"
42150 X=10:Y=07:GOSUB 31000:PRI
      NT"CRITERIO DI ";T$
42160 X=11:Y=07:GOSUB 31000:PRI
      NT"";
42165 LM=L1(SC):GOSUB 46000:C$=
      I$
42166 :
42168 IF RI THEN RETURN
42170 :
42180 OPEN 1,4
42190 PRINT#1,CHR$(15);
42195 IF ST<>0 THEN CLOSE1:GOSU
      B 40500:GOTO 500
42200 GOSUB 42700:GOSUB 42500
42210 PRINT#1,CHR$(27);"(";CHR$
      (0);
42250 FOR I=1 TO NR
42260 ::A$=STR$(I)
42270 ::IF LEN(A$)<3 THEN A$="
      "+A$:GOTO42270
42275 ::A$=MID$(A$,2)
42280 ::IF C$="*"THEN GOSUB 424
      00:GOTO 42300
42290 ::IF MID$(RE$(G(I)),IN(SC
      ),LEN(C$))=C$ THEN GOSUB
       42400:GOTO 42300
42300 NEXT I
42310 PRINT#1:CLOSE 1
42320 XT=13:YT=14:GOSUB 61000
42330 GOTO500
42340 :
42400 REM ** STAMPA RECORD **
42410 PRINT#1,A$;" ";
42420 FOR J=1 TO NC
42430 ::PRINT#1,MID$(RE$(G(I)),
      IN(J),L1(J));" ";
42440 NEXT J
42450 PRINT#1,CHR$(13);
42460 RETURN
42470 :
42500 REM ** STAMPA NOME CAMPI
      **
42510 :
42520 PRINT#1,CHR$(27);"D";CHR$
      (123);CHR$(0);
42550 PRINT#1,CHR$(9);"    SOCI
      O"
42560 PRINT#1,"  ";
42570 FOR K=1 TO 5
42580 I$=CA$(K):LM=-L1(K):GOSUB
       62000
42590 PRINT#1," ";I$;" ";
42600 NEXT K
42610 PRINT#1,"   D.NASC.   FIS
      I   S.C 85/86 ";
42615 PRINT#1
42620 RETURN
42630 :
42700 REM ** STAMPA TESTATA **
42710 PRINT#1,CHR$(27)"G";
```

```
42720 PRINT#1,CHR$(14);:SP$=""
42730 :
42740 LN=LEN(NS$)
42750 NA=(66-LN)/2
42760 FORI=1TO NA:SP$=SP$+"*":N
      EXT
42770 PRINT#1,SP$;NS$;SP$
42780 PRINT#1,CHR$(20)
42790 PRINT#1,CHR$(27)"H"
42800 RETURN
42810 :
42820 :
43000 REM ===================
      ========
43010 REM ===     RICERCA LOGICA
       ====
43020 REM ===================
      ========
43040 :
43050 ST$=" RICERCA LOGICA     "
43060 RI=(1=1):T$="RICERCA":GOS
      UB 42050
43070 REM   ↑ CHIEDE CAMPO E CRI
      TERIO
43080 :
43090 FOR J=1 TO NR
43100 ::IF MID$(RE$(G(J)),IN(SC
      ),LEN(C$))<>C$ THEN 43170
43105 REM ==== RECORD TROVATO =
      ===
43110 ::GOSUB 60000:GOSUB 59000
43115 X=01:Y=13:GOSUB 31000:PRI
      NT"▮ RICERCA LOGICA ▮"
43120 ::FOR K=1 TO NC
43130 ::::X=X1(K):Y=Y1(K):GOSUB
       31000:PRINTMID$(RE$(G(J)
      ),IN(K),L1(K));
43140 ::NEXT K
43150 ::X=21:Y=02:GOSUB 31000:P
      RINT" [F1],[C]ONTINUO RIC
      ERCA,[M]ODIFICA "
43160 ::CA$="MC":GOSUB 45000
43165 ::R=J:ON CH GOTO 43250
43170 NEXT J
43180 REM ==== FINE RECORDS ===
43190 X=10:Y=4:GOSUB 52000
43200 X=13:Y=12:GOSUB 31000:PRI
      NT"FINE ELEMENTI"
43210 XT=17:YT=11:GOSUB 61000
43220 :
43230 GOTO 500
43240 :
43250 X=21:Y=02:GOSUB 31000:PRI
      NT"
          "
43260 GOTO 11160
43270 :
45000 REM ===================
      ========
45010 REM == CONTROLLO TASTO PR
      EMUTO ==
45020 REM ===================
      ========
45025 :
45030 LO=LEN(CA$):IF CA$="**" T
      HEN 45100
45040 POKE198,0:WAIT198,1:GET X
      1$
45045 IF X1$="▮" THEN 500
45050 FOR X9=1 TO LO
45060 IF MID$(CA$,X9,1)=X1$ THE
      N CH=X9:RETURN
45070 NEXT X9
45080 GOTO 45040
45100 POKE198,0:WAIT198,1:GET X
      1$:IF X1$="▮" THEN 500
45110 RETURN
45120 :
46000 REM ===================
```

```
    =========
46010 REM ===   INPUT CONTROLLA
      TO  ===
46020 REM ====================
      =========
46030 I$="":POKE198,0
46040 PRINT"_";
46041 GET X$:IFX$=""THEN46041
46043 IF X$="■" AND LEFT$(ST$,6
      )=" INSER"THEN NR=NR-1:GO
      TO 500: REM  MENU'
46045 IF X$="■" THEN 500 : REM
       MENU'
46050 IF X$="■"ORX$="■"ORX$="■"
      ORX$="■"ORX$="■"ORX$="■"T
      HENPRINT"■";:GOTO46040
46055 IF X$=CHR$(141) THEN PRIN
      T"■";:GOTO 46040:REM [SHI
      FT + <CR>]
46060 IF X$=CHR$(20)ANDLEN(I$)>
      =1THEN I$=LEFT$(I$,LEN(I$
      )-1):PRINT"■ ■■";:GOTO460
      40
46070 IF X$=CHR$(13) AND I$<>""
       THEN PRINT"■ ";:LM=-LM:R
      ETURN
46075 IF X$=CHR$(13) THEN PRINT
      "■";:GOTO 46040
46080 IF LEN(I$)>38-X THEN PRIN
      T"■";:GOTO46040
46085 IF LM>0 AND LEN(I$)>LM-1
       THEN PRINT"■";:GOTO46040

46090 IF X$=CHR$(20) THEN PRINT
      "■";:GOTO46040
46100 I$=I$+X$:PRINT"■";X$;:GOT
      O46040
46110 :
50000 REM **  CONTROLLA PRESEN
      ZA   **
50010 REM **  RECORDS IN MEMOR
      IA   **
50020 :
50030 IF NR<>0 THEN RETURN
50040 X=06:Y=05:GOSUB 52000
50050 X=08:Y=09:GOSUB 31000:PRI
      NT"NESSUN RECORD IN MEMOR
      IA"
50060 XT=13:YT=13:GOSUB 61000
50070 GOTO 500
50080 :
52000 REM ** CREA FINESTRA VIDE
      O **
52010 :
52030 X0=X:GOSUB 31000:PRINT"┌
      ─────────────────────────
      ─────┐";
52040 FOR X=X0+1 TO X0+7:GOSUB
      31000
52050 PRINT"|
                 |";
52060 NEXT
52070 X=X0+8:GOSUB 31000:PRINT"
      └────────────────────────
      ─────┘";
52080 RETURN
59000 :
59010 REM **  STAMPA MASCHERA P
      ER  **
59020 REM ** INPUT O MODIFICA D
      ATI **
59030 :
59040 FOR I=1 TO NC
59050 ::X=X1(I):Y=2:GOSUB 31000

59060 ::PRINTCA$(I);
59070 NEXT I
59080 X=12:Y=00:GOSUB 31000:PRI
      NT"
      ────────────────────────";
59090 X=13:Y=00:GOSUB 31000:PRI
      NT"
      ────────────────────────";
59100 RETURN
59110 :
59500 REM ** CONTROLLO ERRRORE
      DISCO **
59510 :
59520 REM LEGGE ERRORE
59530 INPUT#15,NE,ME$,TE,SE
59540 :
59550 IF NE=0 OR NE=63 THEN RET
      URN
59560 X=06:Y=05:GOSUB 52000
59570 ME$=STR$(NE)+" "+ME$+STR$
      (TE)+" "+STR$(SE):P=(38-L
      EN(ME$))/2
59580 X=09:Y=P+1:GOSUB 31000:PR
      INT ME$
59590 XT=13:YT=12:GOSUB 61000
59600 CLOSE 1
59610 GOTO 20000
60000 REM ** PULISCE SCHERMO **

60010 :
60020 FOR Y1=1 TO 21
60030 X=Y1:Y=1:GOSUB 31000:PRIN
      T"
                               "
60040 NEXT Y1
60050 X=24:Y=0:GOSUB 31000:PRIN
      T"STATO  ===>";ST$;
60060 RETURN
60070 :
61000 REM ** ASPETTA PRESSIONE
      TASTO **

61010 :
61020 X=XT:Y=YT:GOSUB 31000:PRI
      NT" ";R$;"PREMI UN TASTO■
      "
61030 GET A$:IF A$<>"" THEN PRI
      NT"■";:RETURN
61040 FOR DE=0 TO 200: NEXT DE
61050 IF R$="■" THEN R$="■":GOT
      O61020
61060 R$="■":GOTO 61020
61070 :
62000 REM ** ALLINEAMENTO RECOR
      DS **
62010 :
62020 IF LEN(I$)<-LM THEN I$=I$
      +" ":GOTO 62020
62030 RETURN
```

PROTECTOR C-128

Protettore di programmi in Basic ed in L.M.

Di Marco Menichelli

Il Protector C-128 serve a proteggere qualsiasi programma in Basic od in Linguaggio Macchina e a darne il relativo autostart. Una volta protetti, i programmi non possono essere listati o disassemblati. La protezione non crea errori sulle tracce del dischetto e danni al drive in funzione. I programmi protetti possono essere tuttavia copiati su altri dischi, ma solo per mezzo di copiatori del tipo CLONE.

Prima di iniziare a commentare le linee del programma Protector C-128, è necessario chiarire cosa accade al momento dell'accensione del Commodore 128.

Quando si accende il C-128, oppure si preme il pulsante di RESET, viene coinvolta una parte del Sistema Operativo preposta a moltissime funzioni e verifiche. Una di queste ultime, che inizia alla locazione $F88C del banco 15, è quella che controlla se il disk drive è acceso. Se la verifica dà esito positivo, e dentro il drive è stato inserito un dischetto, allora viene caricato nel buffer della cassetta, locato a partire da $0B00, il contenuto del settore zero appartenente alla traccia n.1. A questo punto il Sistema Operativo controlla immediatamente se le prime tre locazioni del buffer della cassetta contengono la sigla «CBM». Avendola trovata, invia allo schermo la stringa «BOOTING» e salva il contenuto delle quattro locazioni, che seguono la sigla, in altre di pagina zero. Quest'ultima operazione riguarda da vicino il funzionamento di Protector.

In un secondo momento il comportamento del S.O. cambia a seconda di ciò che trova in coda a queste locazioni. Se vi sono due zeri consecutivi, viene allora coinvolta una routine in Linguaggio Macchina, che deve trovarsi già scritta a partire dalla locazione successiva ai due zeri. Se invece si trova di fronte un solo zero, esegue l'operazione di caricamento di un file PRG, prima di passare di fronte alla routine di cui sopra, il cui nome è locato nelle posizioni successive allo zero e terminanti ancora con uno zero.

Abbiamo dunque analizzato cosa accade se il S.O. trova uno o due zeri; se non ne identifica nessuno è invece in grado di comprendere di trovarsi di fronte ad un messaggio da inviare allo schermo; lo invia aggiungendo tre punti, controllando l'ultima delle quattro locazioni salvate precedentemente. Normalmente trova ancora 0, ma non nel caso di Protector 128, dal momento che quest'ultimo scrive in partenza le informazioni necessarie a caricare da 1 a 356 blocchi di programmi in Basic o in Linguaggio Macchina senza che la directory ne sia a conoscenza. Pensate un po', 356 blocchi da sfruttare in tutti i loro 256 byte per blocco per un totale di 91.136 byte. Peccato solo che il C-128 non ne possa usare tanti posti in sequenza!

Le prime due locazioni, delle quattro «famose», contengono l'indirizzo iniziale del programma da proteggere, che deve essere caricato in blocchi consecutivi dal S.O. La terza locazione contiene il numero del banco in cui si desidera locare tale programma ed infine la quarta cella contiene il numero dei blocchi da cui è costituito.

Non dovete preoccuparvi se quanto è stato esposto può sembrare troppo complicato, perché Protector C-128 è in grado di svolgere con semplicità

tutte le operazioni, facendo in modo che il programma, non possa essere listato e venga cancellato ogniqualvolta si tenti di interromperlo.

Ci soffermiamo ora ad analizzare il programma in Basic ed in particolare le linee che costituiscono il cuore della protezione.

La linea 120 lavora in coppia con la 820 e la 810, che hanno il compito di segnalare gli eventuali errori e di far ripartire il programma.

La linea 130 controlla il funzionamento del computer in modo 40 oppure 80 colonne ed inizializza la variabile T%, che viene usata per controllare l'output sullo schermo.

Dalla linea 140 alla 210 si ottiene la schermata di presentazione (da cui si esce premendo un tasto qualsiasi) e si inizializzano ad 1 le variabili T e S, indici di traccia e di settore.

Le linee dalla 280 alla 320 presentano il menu con le due opzioni descritte nelle righe stesse.

Le linee dalla 390 alla 410 entrano in esecuzione nel momento in cui si è deciso di proteggere un programma in L.M. e preparano tre delle quattro importanti informazioni menzionate precedentemente.

- L% ed H% contengono l'indirizzo a partire dal quale viene locato il programma.

- B% contiene il numero del banco in cui deve essere riposto.

- SL% ed SH% riportano l'indirizzo di start del programma e servono poi per far partire automaticamente il programma protetto.

Le linee 460 e 470 hanno il compito di richiedere il nome del programma da proteggere, caricandolo a partire dalla fine di Protector C-128 (per poter usufruire di tutto lo spazio disponibile in RAM 0), ricavando questa informazione dalle locazioni $1210 e $1211; ottengono poi l'indirizzo della fine del programma testé caricato dalle locazioni $AE e $AF.

La linea 480 esegue una parte della protezione, modificando ciascun byte con un OR ESCLUSIVO per mezzo della lettera «M» (#$4D).

La linea 510 calcola il numero dei blocchi occorrenti a stivare sul disco il programma, arrotondando il numero per eccesso. La variabile BL%, nell'ordine la quarta rispetto alle quattro menzionate, conterrà tale numero.

A partire dalla linea 570 si inizia il trasferimento dei dati, che costituiscono il programma da proteggere, a partire dal blocco 1 della traccia 1 e seguenti, prelevati con le istruzioni della linea 580. I blocchi e le tracce vengono poi trasferiti nella linea 600.

La linea 650, utile solo se si sta proteggendo un programma in Basic, inizializza le due variabili L% ed H%, già conosciute, con l'indirizzo $1C01, l'inizio di caricamento di tutti i programmi in Basic.

La linea 660 e la linea seguente costituiscono il nucleo centrale del programma di protezione. Si inizia infatti a costruire una stringa, chiamata STRINGA DI BOOT, che reca la sigla «CBM», alla quale vengono aggiunte singolarmente, in ordine, le quattro istruzioni alle quali abbiamo più volte accennato. Viene quindi inserito il messaggio N$, che altro non è che il nome del programma protetto, quindi i due zero che hanno la funzione di indicare al S.O. che non vi è altra operazione da eseguire se non che caricare in memoria i blocchi indicati e di lasciare la gestione del programma alla routine in L.M. che li segue.

A questo punto, tralasciando di commentare la linea 660, è bene aprire una parentesi sui codici del L.M. contenuti nella stringa di BOOT.

Apparentemente, infatti, questi codici non hanno nessun significato. Se provassimo ad inserirli sequenzialmente in RAM e poi andassimo a leggerli con un disassemblatore, non ci si troverebbe di fronte ad un programma in L.M., ma ad una serie di punti interrogativi e ad una qualche sporadica istruzione. In realtà, invece, le istruzioni sono presenti, la CPU lavora correttamente ed il programma segue le normali procedure di funzionamento.

La chiave è racchiusa in alcuni codici della linea 890, che non sono riconosciuti da nessun disassemblatore. Gli altri, contenuti nelle linee DATA dalla 900 in poi, sono stati codificati e costituiscono due distinti programmi, che saranno caricati nella stringa di BOOT a seconda dell'opzione scelta. Chi si occupa di decodificarli sono le «istruzioni fantasma» della linea 890, che lavorano nel modo seguente:

```
LDA  $03          Carica A con il contenuto di
                  $03 (= # $0B)

STA  $B3
LDA  # $00        e crea l'indir. $0B00
                  collocandolo in $B2 e $B3

STA  $B2
LDY  # $3D        Carica Y con l'offset per
                  l'inizio della decod.

SEC               Pone 1 nel carry loop
LDA  ($B2), Y     Inizia a raddoppiare, con
                  riporto

ADC  ($B2), Y     ogni contenuto delle
                  locazioni

STA  ($B2), Y     a partire da $0B3D
INY               fino alla locazione $0BFF.
BNE  loop
```

I dati erano stati naturalmente codificati precedentemente seguendo il procedimento inverso, cioè con una divisione con carry a 0 e a partire dall'ultima istruzione.

Le «istruzioni fantasma» che operano in maniera analoga a quelle conosciute sono le seguenti:

Nome	Codice	Mnemonico	Descrizione
AXL	A7	AXL $03	Carica A ed X con il contenuto dell'operando.
SAX	87	SAX $B3	Pone nella locaz. indicata dall'operando il risultato dell'AND fra A e X.
ANL	4B	ANL #$B1	Esegue un AND fra A e operando, poi un LSR. Il risultato si trova in A. (nel nostro caso lo scopo è azzerare A e di settare il carry)
SAX	87	SAX $B2	Come per il SAX precedente.
LDY	A0	LDY #$3D	Codice sconosciuto.
DPM	33	DPM ($B2), Y	Raddoppia il contenuto indicato dell'indir. contenuto in operando.
INY	C8	INY	Codici sconosciuti.
BNE	D0	BNE $loop · DPM	

Naturalmente, per confondere anche i codici noti, si è reso necessario aggiungere altre «istruzioni fantasma» inattive che effettuano salti di uno o due byte denominati NOP2 e NOP3:

NOP3 → DC, 5C, 3C, 1C.
NOP2 → 80, 34.

Gli altri caratteri, che seguono i NOP, compongono il cognome dell'autore di Protector e non hanno altra funzione che mettere in difficoltà l'utente.

Nella linea 660 (CHR$ (220)), dopo i due zero, troviamo un NOP3 (CHR$ (220)), seguito dalle due variabili SL% ed SH%, che contengono 0, se si è scelto di proteggere un programma in Basic, altrimenti contengono l'indirizzo dello start del programma in L.M. ricevuto alla linea 410. La linea 670 provvede a completare la stringa di BOOT con i codici dei programmi in L.M., sulla cui struttura sarà opportuno soffermarsi in seguito, coadiuvata dalle linee 840, 850, 860 e 870. Nelle linee 740 - 760 si completa il programma Protector C-128 con la scrittura della stringa di BOOT nel settore 0 della traccia 1.

Chiariamo adesso brevemente in che modo i programmi in L.M. codificati, dopo la loro decodifica da parte delle «istruzioni fantasma», completino la protezione:

1) inibiscono il comando RUN / STOP - RESTORE.

2) modificano i puntatori di una routine del S.O. usata all'accensione ed al reset, che inizia a F$E1F0 il cui compito è quello di controllare nel banco 1, se esiste ancora una sigla CBM per il salto incondizionato all'indirizzo specificato nei due codici successivi alla sigla stessa. La modifica indirizza il S.O. verso una routine che provvede a cancellare il programma protetto. La sigla si trova alle locazioni 1$FFF5, 1$FFF6 e 1$FFF7; l'indirizzo di salto alle locazioni 1$FFF8 e 1$FFF9.

3) intercettano i puntatori della routine di ERROR e READY per indirizzare il S.O. alla routine di cancellazione del programma protetto.

4) decodificano il programma con un OR Esclusivo con la lettera «M», operando in maniera analoga alla codificazione della linea Basic 480.

5) danno lo start al programma protetto in L.M. utilizzando la routine JMPFAR locata a $02E3.

6) attivano il programma in Basic protetto utilizzando le subroutine del S.O. di cui:

JSR	$4F4F	riconcatena le linee Basic.
JSR	$4F82	aggiorna i puntatori di fine programma $1210 e $1211.
JMP	$5AAF	dà il RUN al programma Basic.

7) cancellano il programma protetto, sia che si prema il tasto di RESET soltanto, sia che si utilizzi anche il tasto RUN/STOP.

8) obbligano il S.O. a riposizionare i puntatori corretti della subroutine accennata al punto 2.

9) danno il RESET al computer con la routine $FFD3 oppure escono dal programma che è già stato cancellato.

Prima di proteggere un programma in Basic è necessario sapere che è utile usare l'istruzione TRAP all'interno del programma stesso in quanto, avendo Protector C-128 intercettato la routine di ERROR, qualsiasi errore all'interno delle linee comporta la cancellazione del programma in memoria ed il conseguente ritorno al READY. Chi invece programma in L.M., dovrà porre particolare attenzione a non modificare gli indirizzi che si trovano nelle locazioni $0300 e $0301, $0328 e $0329 della RAM COMUNE e da $FFF5 a $FFF9 compresa del BANCO 1. Si rende inoltre necessario non usare assolutamente la zona RAM che va da $0B00 a $0BFC in quanto indispensabile al

programma di protezione.

Prima di procedere all'uso di Protector C-128 è bene chiarire che il dischetto, che contiene il programma protetto, perde irrimediabilmente il contenuto del blocco 0 in traccia 1 ed in più quello di tanti blocchi quanti sono necessari a caricare il programma. Le tracce da 1 a 17 vengono interessate anche da altri programmi caricati con un normale comando di SAVE, partendo dalla 17 ed andando verso la 1, seguendo il progressivo inserimento dei dati nel dischetto.

È forse superfluo accennare al fatto che si può proteggere un solo programma per dischetto e che è opportuno usare la protezione di un programma su un dischetto appena formattato. Tuttavia il Protector C-128 aggiorna anche la BAM e non si corre l'opposto pericolo di caricare un programma qualsiasi insieme a quello protetto: questo fintanto che non si decida di riorganizzare la BAM con i comandi di VALIDATE o COLLECT.

Usando il drive 1571, in modo 1571, si può verificare, durante la protezione di un programma, un inspiegabile errore del tipo «DEVICE NOT PRESENT». Per ovviare a ciò è sufficiente attivare, precedentemente, la stampante oppure far funzionare il 1571 in modo 1541, digitando in diretta OPEN 15,8,15 «UO<MO».

L'unica via per caricare un programma protetto con Protector C-128 è quella di inserire il dischetto che lo contiene nel drive acceso e poi di attivare in un secondo momento il computer, oppure, se questo è già in funzione, è necessario premere il tasto di RESET.

Listato. *Protector C-128*

```
10 REM  ********************************
20 REM  **                            **
30 REM  **     PROGRAM-PROTECTOR      **
40 REM  **                            **
50 REM  **       PER C=128  DI        **
60 REM  **                            **
70 REM  **     MENICHELLI MARCO       **
80 REM  **                            **
90 REM  ********************************
100 :
110 :
120 TRAP820
130 IFRGR(0)=5THENT%=20:FAST:ELSECOLOR0,5:COL
    OR1,5:COLOR4,5:COLOR5,14
140 WINDOW0,0,2*T%+39,24,1:T=1:S=1:FORX=1TO16:
    SP$=SP$+" ":NEXT:A$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
150 FORX=1TO24:PRINTLEFT$(A$,T%*2+40);:NEXT
160 WINDOWT%+8,9,T%+30,14,1
170 CHAR,1,1,"┌──────────────────┐"
180 CHAR,1,2,"|  PROTECTOR-C=128  |"
190 CHAR,1,3,"| (C) M. MENICHELLI |"
200 CHAR,1,4,"└──────────────────┘"
210 GETKEYA$:WINDOW0,0,2*T%+39,24,1
220 :
230 REM **** MENU ****
240 :
250 CHAR,T%,1,"     PROTECTOR C=128 (C) M.MENI
    CHELLI    ",1:WINDOW0,3,2*T%+39,24
260 CHAR,0,5,"INSERISCI UN DISCHETTO FORMATTA
    TO  NEL  DRIVE E .................. PREMI
    SPAZIO.",1:GETA$:IFA$<>" "THEN260:ELSEDCL
    EAR:SCNCLR:A=1
270 PRINTCHR$(15);:CHAR,T%+2,3,"SCEGLI CON CR
    SR DOWN E PREMI RETURN":PRINTCHR$(143)
280 A$="":DOWHILEA$<>CHR$(13)
290 CHAR,T%+7,8,"PER UN PROGRAMMA IN L.M.",-
    (A=1)
300 CHAR,T%+7,11,"PER UN PROGRAMMA IN BASIC",
    -(A=2)
310 GETA$:IFA$=CHR$(17)THENA=A+1:A=A+(A>2)*2
320 LOOP:SCNCLR
330 :
340 ONAGOTO380,450
350 :
360 REM **** OPZIONE 1 ****
370 :
380 GOSUB450
390 CHAR,0,11,"":INPUT"BANCO DESTINAZIONE (HE
    X) ";B$:B%=DEC(B$):IFB%<0ORB%>15THEN390
400 CHAR,0,13,"":INPUT"LOCAZIONE  INIZIO  (HEX)
    ";LI$:LI=DEC(LI$):H%=LI/256:L%=LI-H%*256
410 CHAR,0,15,"":INPUT"LOCAZIONE  START  (HE
    X) ";SS$:SS=DEC(SS$):SH%=SS/256:SL% =SS-S
    H%*256:GOTO660
420 :
430 REM **** OPZIONE 2 ****
440 :
450 PRINTCHR$(7):CHAR,0,2,"INSERISCI   IL  DI
    SCO   CONTENENTE  IL  PROGRAMMA E .......
    ..... PREMI UN TASTO.",1:GETKEYA$:SCNCLR
460 CHAR,0,1,"":INPUT"NOME DEL PROGRAMMA ";N$
    :BANK0:H%=PEEK(DEC("1211")):L%=PEEK (DEC("
    1210")):BANK15:LI=H%*256+L%
470 CHAR,0,3,"LOADING "+N$,1:BLOAD(N$),B0,P(L
    I):LF=PEEK(DEC("AF"))*256+PEEK(DEC("AE"))
    :N$=RIGHT$(SP$+N$,16)
480 CHAR,T%+2,6,"ATTENDERE PREGO, LO STO CODI
    FICANDO":BANK0:FORX=LITOLF:POKEX,XOR(PEEK
    (X),77):NEXT:SLEEP1:BANK15
490 SCNCLR:PRINTCHR$(7):CHAR,0,1,"INSERISCI
    IL  DISCO   DESTINAZIONE  E ...........
    ....... PREMI UN TASTO.",1:GETKEYA$:SCNCLR
500 OPEN15,8,15,"I":GOSUB800:OPEN13,8,13,"#":
    GOSUB800
510 LT=LF-LI:BL%=LT/256:B=LT-BL%*256:IFBTHENB
    L%=BL%+1
520 CHAR,T%+2,4,"ATTENDERE PER FAVORE, STO SC
    RIVENDO",1:CHAR,T%+12,6,"TRACCIA  SETTORE"
530 :
540 REM **** SCRITTURA DEL BLOCCO N.1 NELLA T
    RACCIA N.1 E SEGUENTI ****
550 :
560 FORX=1TOBL%:A$=RIGHT$(" "+STR$(T),2)+LEFT
    $(SP$,6)+RIGHT$(" "+STR$(S),2):CHAR,T%+15
    ,8,A$
570 PRINT#15,"B-F:"0;T;S:PRINT#15,"B-A:"0;T;S
    :PRINT#15,"B-P:13,0"
580 BANK0:FORI=0TO255:N%=PEEK(LI):PRINT#13,CH
    R$(N%);:LI=LI+1:NEXT
590 PRINT#15,"B-P:13,0":PRINT#15,"U2:"13;0;T;
    S:GOSUB800
600 S=S+1:IFS>20THENS=0:T=T+1
610 NEXT:BANK15:IFA=1THENRETURN
620 :
630 REM **** STRINGA DI BOOT ****
640 :
650 L%=DEC("01"):H%=DEC("1C")
660 BT$="CBM"+CHR$(L%)+CHR$(H%)+CHR$(B%)+CHR$
    (BL%)+N$+CHR$(0)+CHR$(0)+CHR$(220)+ CHR$(S
    L%)+CHR$(SH%)
```

```
670 GOSUB840:IFA=1THENGOSUB840:ELSERESTORE940
    :GOSUB840
680 IFA=1ANDCL<>27346THENCHAR,0,19,"ERRORE NE
    I DATA DEL L.M.":DCLOSE:END
690 IFA=2ANDCL<>25636THENCHAR,0,19,"ERRORE NE
    I DATA DEL BASIC":DCLOSE:END
700 :
710 REM **** SCRITTURA DEL BLOCCO N.0 NELLA T
    RACCIA N.1 ****
720 :
730 CHAR,T%+15,8,"1          0   "
740 PRINT#15,"B-F:0,1,0":PRINT#15,"B-A:0,1,0"
    :PRINT#15,"B-P:13,0"
750 PRINT#13,BT$;:GOSUB800
760 PRINT#15,"B-P:13,0":PRINT#15,"U2:13,0,1,0"
    :CHAR,0,17,N$+" E' PROTETTO",1:GOTO810
770 :
780 REM **** SUBROUTINE CONTROLLO ERRORE DISC
    O ****
790 :
800 IFDS=0THENRETURN
810 CHAR,0,19,DS$:BANK15:DCLOSE:SLOW:SLEEP3:
    RUN
820 CHAR,0,18,ERR$(ER),1:GOTO810
830 :
840 READA$
850 DOWHILEA$<>"*"
860 BY=DEC(A$):BT$=BT$+CHR$(BY):CL=CL+BY
870 READA$:LOOP:RETURN
880 :
890 DATA 5C,4D,45,A7,03,87,B3,4B,B1,87,B2,3C,
    4E,49,A0,3D,33,B2,80,4C,1C,43,48,C8,DC,45
    ,4C,34,4C,D0,F1,34,49,*
900 DATA D4,FC,46,94,81,D4,85,C6,94,81,54,FC,
    C2,E1,D4,FF,42,E2,D4,E1,C6,5C,01,50,80,54
    ,D8,46,80,01,D1,3F,90,57,01,E4,D4,85,C6,
    80,01,D1,3F,90,57,81,56,83,85
910 DATA C2,FD,D6,82,05,D7,81,05,56,82,85,42,
    7E,C3,7E,42,FF,D4,FE,46,55,81,C6,5C,01,50
    ,00,53,7E,90,E9,FB,A4,26,53,7E,10,ED,7B,
    64,E8,78,73,7F,E3,7D,E8,F5,D6
920 DATA 82,05,D7,8D,05,56,8D,85,42,01,C3,01,
    42,02,A6,71,01,D7,81,05,56,82,05,C3,61,42,
    E2,D4,E1,C6,5C,01,50,80,54,00,D7,82,05,10,
    ED,7B,64,E8,7A,73,62,67,83
930 DATA 05,68,77,10,12,F1,D4,1F,D1,A6,46,80,
    01,C7,80,01,50,00,CC,4C,80,05,64,E0,71,48
    ,7C,04,B4,14,02,E8,02,51,40,A6,9F,26,A6,9
    E,FF,D4,7F,1C,30,*
940 DATA D4,F6,46,94,81,D4,85,C6,94,81,54,FC,
    C2,E1,D4,FF,42,E2,D4,E1,C6,5C,01,50,80,54
    ,CF,46,80,01,D1,3F,90,57,01,E4,D4,85,C6,8
    0,01,D1,3F,90,57,81,56,83,05
950 DATA D7,81,05,56,82,85,C6,CF,05,C3,7E,42,
    FF,D4,9F,46,80,7F,50,80,D8,FE,A4,A6,C8,7E
    ,64,E8,7B,73,7F,67,83,05,68,78,46,80,7F,9
    0,A7,27,10,C1,27,26,53,2D,51
960 DATA 00,C3,FE,52,97,42,FF,D4,96,C6,5C,01,
    50,80,54,00,55,10,ED,7B,64,E8,7B,73,17,E3
    ,7E,E8,78,CC,46,08,89,C6,08,89,52,FF,42,1
    7,10,12,71,D1,1F,D0,26,47,80
970 DATA 01,C6,80,01,50,00,CC,4C,80,05,64,E0,
    6B,48,7C,04,B4,14,02,E8,02,51,40,A6,9F,26
    ,A6,9E,FF,D4,7F,1C,30,*
```

Disassemblato di PROTECTOR C-128 - Routine protezione programmi in Basic

Nome	Codice Mnemonico		Descrizione
0B3D	LDA	#$ED	
0B3F	STA	$0328	Intercetta il vettore di STOP per
0B42	LDA	#$0B	
0B44	STA	$0329	inibire i tasti di RUN/STOP e RESTORE.
0B47	LDA	#$F8	Prepara i puntatori $C3 e $C4 per la routine di STASH
0B49	STA	$C3	
0B4B	LDA	#$FF	con l'indirizzo $FFF8.
0B4D	STA	$C4	
0B4F	LDA	#$C3	
0B51	STA	$02B9	Prepara STA ($C3), Y.
0B54	LDY	#$00	
0B56	LDA	#$9E	Carica in A il byte basso dell'indirizzo $0B9E.
0B58	STA	$0300	Modifica il vettore LO di ERROR.
0B5B	LDX	#$7F	Inizializza X con il codice del BANCO 1.
0B5D	JSR	$02AF	STASH, pone in 1$FFF8 il contenuto di A.
0B60	INY		
0B61	LDA	#$0B	
0B63	STA	$0301	Modifica il vettore HI di ERROR.
0B66	LDX	#$7F	Pone in 1$FFF9 il byte alto dell'indirizzo $0B9E
0B68	JSR	$02AF	tramite la subroutine di STASH.
0B6B	LDA	$0B06	Numero dei blocchi.
0B6E	LDX	$0B03	Byte L0 indirizzo inizio programma.
0B71	LDY	$0B04	Byte HI indirizzo inizio programma.
0B74	STA	$0B9F	Colloca in 0B9F il numero dei blocchi per completare l'istruz. in riga 0B9E.
0B77	STX	$FD	Inizializza i puntatori $FD e $FE all'inizio
0B79	STY	$FE	del programma in BASIC.
0B7B	LDA	#$3F	Inizializza con il codice #$3F
0B7D	STA	$FF00	la MMU al fine di lavorare in Banco 0.
0B80	LDY	#$00	Decodifica il
0B82	LDA	($FD),Y	programma protetto
0B84	EOR	#$4D	
0B86	STA	($FD),Y	con il carattere chiave #$4D
0B88	INY		
0B89	BNE	$0B82	
0B8B	INC	$FE	
0B8D	DEC	$0B06	fino a che non si giunge alla fine.
0B90	BNE	$0B82	
0B92	STY	$FF00	Codice 0 nella MMU per tornare al Banco 15.

0B95	JSR	$4F4F	Riorganizza i puntatori delle linee BASIC.
0B98	JSR	$4F82	Aggiorna i puntatori di fine programma.
0B9B	JMP	$5AA6	Dà il RUN al programma in BASIC.

Inizio della routine che cancella il programma protetto al momento di un reset, di un errore o per fine programma. Viene attivata dalla routine del S.O. che inizia da F$E1F0 oppure da F$4D3C.

0B9E	LDX	#$	Riceverà il numero dei blocchi dalla riga 0B74.
0BA0	STX	$FD	Contatore del numero dei blocchi di 256 byte.
0BA2	LDA	$2E	Salva il contenuto del puntatore alto all'inizio programma nella locazione $FE.
0BA4	STA	$FE	
0BA6	LDA	#$2D	Prepara STA ($2D), Y
0BA8	STA	$02B9	nella routine di STASH. $2D e $2E sono i puntatori
0BAB	LDY	#$00	di inizio di un programma in BASIC, anche rilocato.
0BAD	LDA	#$00	
0BAF	TAX		
0BB0	JSR	$F7DA	Subroutine del S.O. che trasforma il numero del Banco in codice e poi salta alla STASH.
0BB3	INY		
0BB4	BNE	$0BAD	
0BB6	INC	$2E	
0BB8	DEC	$FD	Azzera tutte le locazioni in Banco 0 fino all'ultimo
0BBA	BNE	$0BAD	blocco
0BBC	TYA		
0BBD	STA	$1210	Azzera i puntatori di fine programma che aveva aggiornato
0BC0	STA	$1211	precedentemente con JSR $4F82.
0BC3	LDA	$FE	Ripristina il puntatore di inizio programma con
0BC5	STA	$2E	l'indirizzo HI salvato prima.
0BC7	JSR	$E224	Routine del S.O. che ripristina i puntatori delle due locazioni modificate precedentemente alle linee 0B5D e 0B68.
0BCA	LDX	$# 3F	Ripristina i vettori originali della routine
0BCC	LDY	$#4D	
0BCE	STX	$0300	di ERROR.
0BD1	STY	$0301	
0BD4	LDY	#$00	
0BD6	TYA		Cancella l'intero programma
0BD7	STA	$0B00,Y	
0BDA	INY		mettendo a 0 tutte le locazioni da $0B00
0BDB	CPY	#$D7	
0BDD	BCC	$0BD7	fino a $0BD6
0BDF	PHP		Mette nell'area STACK il registro di stato.
0BE0	PLA		Carica l'Acc. con il valore del registro di stato.
0BE1	AND	#$04	Controlla se il bit dell'INTERRUPT è settato a 1.
0BE3	BNE	$0BEA	Se è vero significa che è stato premuto il pulsante di RESET quindi salta alla routine di RESET,
0BE5	LDX	#80	altrimenti carica in X il codice di NESSUN ERRORE e
0BE7	JMP	$4D3F	va alla routine di READY del S.O.
0BEA	JMP	$FF3D	Routine di RESET del S.O.
			Falsa subroutine di STOP.
0BED	LDA	#$FF	Ignora la lettura della locazione $91 che contiene il flag di STOP.
0BEF	SEC		Simula la comparazione con # $7F.
0BF0	RTS		Termina la routine di STOP.

Disassemblato di PROTECTOR C-128
Routine protezione programmi in LINGUAGGIO MACCHINA

0B3D	LDA	#$F9	Intercetta il vettore di STOP per
0B3F	STA	$0328	
0B42	LDA	#$0B	inibire i tasti di RUN/STOP e RESTORE.
0B44	STA	$0329	
0B47	LDA	#$F8	Prepara i puntatori $C3 e $C4 per la routine di STASH
0B49	STA	$C3	
0B4B	LDA	#$FF	con l'indirizzo $FFF8.
0B4D	STA	$C4	
0B4F	LDA	#$C3	
0B51	STA	$02B9	Prepara STA ($C3), Y.
0B54	LDY	#$00	
0B56	LDA	#$B0	Carica in A il byte basso dell'indirizzo $0BB0.
0B58	STA	$0300	Modifica il vettore LO di ERROR.

0B5B	LDX	#$7F	Inizializza X con il codice del BANCO 1.
0B5D	JSR	$02AF	STASH, pone in 1$FFF8 il contenuto di A.
0b60	INY	1$FFF8	
0B61	LDA	#$0B	
0B63	STA	$0301	Modifica il vettore Hi di ERROR.
0B66	LDX	#$7F	Pone in 1$FFF9 il byte alto dell'indirizzo $0BB0
0B68	JSR	$02AF	tramite la subroutine di STASH.
0B6B	LDA	$0B06	Carica il numero dei blocchi componenti il
0B6E	STA	$FB	programma e lo salva nel contatore $FB.
0B70	LDA	$0B05	Numero del BANCO.
0B73	LDX	$0B03	Indirizzo L0 di inizio programma.
0B76	LDY	$0B04	Indirizzo HI di inizio programma.
0B79	STA	$FC	
0B7B	STX	$FD	
0B7D	STY	$FE	
0B7F	LDA	#$FD	
0B81	STA	$02AA	Prepara LDA ($FD), Y FETCH
0B84	STA	$2B9	e STA ($FD), Y STASH.
0B87	LDY	#$00	
0B89	LDX	$FC	Carica in X il numero del Banco.
0B8B	JSR	$F7D3	Subroutine del S.O. che trasforma il num. del banco in codice e salta alla FETCH.
0B8E	EOR	#$4D	Decodifica con #4D «M» il contenuto di A.
0B90	LDX	$FC	
0B92	JSR	$F7DA	Subroutine del S.O. che trasforma il num. del banco in codice e salta alla STASH.
0B95	INY		
0B96	BNE	$0B89	Continua per 256 byte.
0B98	INC	$FE	Incrementa il puntatore HI al programma.
0B9A	DEC	$FB	Decrementa il cont. del numero dei blocchi.
0B9C	BNE	$0B89	Continua se ci sono ancora byte da decodificare.
0B9E	LDA	$0B05	Preleva il numero del BANCO.
0BA1	LDX	$0B1B	Preleva l'indirizzo HI di start del programma.
0BA4	LDY	$0B1A	Preleva l'indirizzzo L0 di start del programma.
0BA7	STA	$02	Aggiorna i puntatori
0BA9	STX	$03	della routine
0BAB	STY	$04	di JMPFAR.
0BAD	JMP	$02E3	JMPFAR, dà lo start al programma protetto.

Inizio della routine che cancella il programma protetto al momento di un reset di un errore o per fine programma. Viene attivata dalla routine del S.O. che inizia da F$E1F0 oppure da F$4D3C.

0BB0	LDX	$0B03	Prepara la routine di STASH
0BB3	LDY	$0B04	
0BB6	STX	$C3	ed i suoi puntatori
0BB8	STY	$C4	
0BBA	LDA	#$C3	
0BBC	STA	$02B9	con l'indirizzo di inizio programma
0BBF	LDY	#$00	
0BC1	LDA	#$00	e lo cancella ponendo 0
0BC3	LDX	$0B05	
0BC6	JSR	$F7DA	
0BC9	INY		in tutte le locazioni
0BCA	BNE	$0BC1	
0BCC	INC	$C4	
0BCE	DEC	$0B06	occupate dal programma protetto.
0BD1	BNE	$0BC1	
0BD3	JSR	$E224	Routine del S.O. che ripristina i puntatori delle sue locazioni modificate precedentemente alle linee 0B5D e 0B68.
0BD6	LDA	#$3F	Ripristina i vettori originali della
0BD8	LDX	#$4D	
0BDA	STA	$0300	routine di ERROR.
0BDD	STX	$0301	
0BE0	LDY	#$00	
0BE2	TYA		Cancella l'intero programma
0BE3	STA	$0B00,Y	
0BE6	INY		mettendo a 0 tutte le locazioni da $0B00 a $0BE2.
0BE7	CPY	#$E3	
0BE9	BCC	$0BE3	
0BEB	PHP		Pone in STACK il registro di stato.
0BEC	PLA		Carica l'Acc. con il registro di stato.
0BED	AND	#$04	Controlla se il bit dell'INTERRUPT è settato a 1.
0BEF	BNE	$0BF6	Se è vero significa che è stato premuto il pulsante di RESET quindi salta alla routine di RESET,
0BF1	LDX	#$80	altrimenti carica in X il codice #$80 = NESSUN ERRORE E
0BF3	JMP	$4D3F	va alla routine di READY del S.O.
0BF6	JMP	$FF3D	Routine di RESET del S.O.
			Falsa subroutine di STOP.
0BF9	LDA	#$FF	Ignora la lettura della locazione $91 che contiene il flag di STOP.
0BFB	SEC		Simula la comparazione con #$7F.
0BFC	RTS		Termina la routine di STOP.

GEOS
Cancel
NO
OK
YES
MICHELE '87

Le chiavi del regno

Un accurato esame del sistema operativo GEOS per svelarne tutti i segreti.

Seconda parte

Di Luca Giachino

Nota dell'autore: tutti gli argomenti e le anticipazioni trattate in questo articolo costituiscono un lavoro di ricerca svolto in completa assenza di manualistica e documentazioni tecniche. I listati dei codici pubblicati sono tratti dai codici originali di GEOS e sono coperti da Copyright by Berkeley Softworks.

Introduzione

Quando ho avuto modo di visionare GEOS per la prima volta, già i pochi elementi a mia disposizione erano sufficienti per stimolare ricerche più approfondite. Da buon conoscitore dei migliori pacchetti applicativi in commercio per il Commodore 64, non mi sono voluto far ingannare dalla accattivante veste estetica del programma, e ho cercato di analizzarlo il più razionalmente e obiettivamente possibile. Ma in breve tempo le apparenze si sono trasformate in qualità di rilievo e il desiderio di conoscere in profondo interesse. Immedesimandomi nei panni dell'utente ho potuto apprezzare la semplicità e la chiarezza con cui il sistema dialoga, la velocità di esecuzione dei molteplici comandi, alcuni inediti per il C-64, la qualità grafica ed estetica e tutte le caratteristiche di cui ho già fatto ampia menzione nel precedente articolo. In qualità di programmatore mi sono spesso interrogato per cercare di spiegarmi quali algoritmi e procedure si alternino dietro le quinte della scena offerta da GEOS. Uno fra i modi migliori per imparare a programmare in linguaggio macchina, consci della velocità e delle potenzialità che può offrire, consiste nello «sviscerare» i programmi di cui si dispone, per apprendere i mille trucchetti che devono far parte del bagaglio di un buon programmatore in L.M. GEOS si è rilevato il programma più interessante con cui abbia mai avuto a che fare. Esso racchiude un vero condensato di nozioni che spaziano dalla grafica alla gestione delle porte e delle periferiche, dalla computazione matematica alla gestione ottimizzata della memoria. Con GEOS non si soddisfano solo delle curiosità per aumentare la familiarità con il microprocessore 6502, ma si gettano le basi per programmare con più facilità e velocità, utilizzando delle routine che GEOS stesso mette a disposizione. In questo articolo il mio sforzo è quello di offrirvi la possibilità di conoscere il mondo di GEOS non solo come utenti, ma anche come «addetti ai lavori».

Per «penetrare» nella struttura di GEOS ho dovuto affrontare tutta una serie di problemi di difficile soluzione. Alcuni nascono dalla evidente complessità del sistema (mi sono infatti sempre chiesto se sia più difficile programmare o analizzare un programma scritto da altri), ma la maggior parte, almeno negli stadi iniziali, sono volutamente stati inseriti dai programmatori della Berkeley Softworks per aumentare le difficoltà di analisi dei codici (escludo da questo ambito la protezione del disco). Certo è che anche sotto questo aspetto mi devo sinceramente congratulare per la fantasia e le capacità dimostrate, tanto è stato il tempo richiesto per analizzare le piccole astuzie adottate per scoraggiare qualunque tentativo di analisi. Gli strumenti di cui mi sono servito

sono fondamentalmente tre: due monitor, di cui uno residente in una cartuccia Fastload (reperibile anche sotto il nome Turboload o simili) ed uno caricabile da disco, denominato Hesmon, ed infine un software molto potente per programmare direttamente su disco, il famoso Disk Maintenance. Il monitor residente nella cartuccia è molto semplice, al punto da non permettere di assemblare utilizzando le pseudo-istruzioni previste dal 6502, ma presenta nel contempo delle caratteristiche decisamente uniche e molto utili. Particolare degno di nota è che non risiede in nessuna zona ram del C-64 e quindi si può considerare completamente trasparente. A differenza dei normali monitor in commercio, non fa uso di nessuna routine contenuta nel Kernal o nell'interprete Basic. È quindi, e per fortuna, completamente indifferente alla configurazione rom/ram del C-64! In altri termini, settando la locazione $01 di pagina zero a # $30 anziché # $37, ottenendo così 64k di ram disponibile, il monitor continua ad operare egregiamente senza che siano stati trasferiti, come di consueto in operazioni di questo genere, né il Kernal né l'interprete nelle sottostanti ram. Questa caratteristica diviene indispensabile per accedere ai codici di GEOS. Oltre a non consentire l'assemblaggio, questo monitor non è in grado di accedere alla stampante. Quindi, per assemblare, esistono in pratica due sistemi: o si imparano a memoria i codici esadecimali delle istruzioni e si digitano direttamente in memoria esplorandola col comando «*», o si ricorre addirittura ad un altro monitor. Hesmon, l'altro monitor utilizzato, oltre a permettere di assemblare, è in grado di accedere alla stampante, particolare importante per l'analisi intrapresa. Ecco spiegata la necessità di due monitor. Il terzo programma si è reso utilissimo per effettuare piccoli cambiamenti ai codici di GEOS, esplorarne i file, con la possibilità di disassemblarli e assemblarli direttamente sul disco, scrivere qualche piccolo programma, sulla falsariga di quello annunciato nell'articolo precedente, direttamente su disco con notevole risparmio di tempo. Esiste inoltre uno strumento di cui un programmatore che si rispetti dovrebbe far uso con frequenza: il tasto di reset. Non è residente nel C-64, ma potete tranquillamente comprarne uno o realizzarlo voi stessi. È molto utile in quanto vi consente di resettare il sistema senza togliere alimentazione alle memorie. Il reset, infatti, riorganizza la memoria da $0000 a $0803, azzerando gli ultimi tre byte che sono i primi della zona riservata ai programmi Basic, e da $D000 a $DDFF, ma non effettua nessun cambiamento nella ram restante, la lascia cioè inalterata. Ogni programmatore ovviamente ha le proprie abitudini, i propri ferri del mestiere, ma se volete veramente penetrare nei meandri di GEOS vi accorgerete che gli strumenti che ho utilizzato si rivelano fondamentali.

Il primo argomento che tratterò sarà un'analisi completa del programma «Geos Boot», quale fondamentale trampolino di lancio, mettendovi in guardia dalle insidie che presenta. In esso è contenuto il primo turbo di cui GEOS fa uso per caricare «Geos Kernal». Questo programma, atto a velocizzare notevolmente il booting del sistema, è esclusivamente utilizzato per «Geos Kernal», e non si tratta della versione finale che andrà poi a risiedere con fissa dimora nel disk-drive. Ma non è in verità molto diverso come protocollo di trasmissione e quindi sarà opportuno analizzarlo nei dettagli, un'occasione in più per toccare con mano un turbo, peraltro molto veloce e ben fatto.

Quando «Geos Boot» cede il controllo a «Geos Kernal», il nostro interesse si sposta finalmente ai codici veri e propri di GEOS. Illustreremo quindi l'accesso ai file, la gestione grafica degli output, finestre comprese, ed il controllo del mouse. Il tutto sarà corredato da esempi e metodi pratici di verifica. Per esempio, la possibilità di resettare il C-64, esplorare e toccare i codici ancora residenti in memoria e richiamare GEOS con poche operazioni, osservando in tempo reale i risultati ottenuti.

Come avevo già avuto modo di sottolineare, il mio proposito è quello di divulgare più informazioni possibili sulla struttura di GEOS, in modo da stimolare la creatività intorno a questo nuovo ambiente di lavoro. La Berkeley Softworks ci ha fornito il nuovo standard per programmare con più facilità e migliori risultati, a noi spetta usarlo. Chiunque volesse inoltre farci pervenire materiale di interesse inerente a GEOS, troverà in noi la massima disponibilità.

Molto spesso nel corso dell'articolo si renderà opportuno addentrarsi nei codici di GEOS spiegandone ogni istruzione in linguaggio macchina. Come dato essenziale assumerò che il lettore non sia digiuno di L.M. e mi soffermerò a spiegare solo le istruzioni del microprocessore 6502 meno note; chi a digiuno fosse veramente può comunque sfruttare quanto dirò per approfondire, coadiuvato possibilmente da un buon manuale sul 6502, il mondo della programmazione in L.M. (perché di un «mondo» si tratta).

Utilizzerò nel corso dell'articolo una terminologia che ho già adottato nella precedente occasione per analizzare la gestione dei file secondo lo standard GEOS. Riferimenti più precisi si possono ottenere consultando il numero scorso di Commodore Gazette.

Prima di dare inizio all'«avventura» che ci porterà a esplorare da vicino il sistema operativo GEOS, mi devo ancora congratulare con la Berkeley Softworks. Il loro lavoro, e voi lettori ve ne renderete conto quando inizierete a manipolarlo, si discosta molto per originalità e complessità dai prodotti realizzati normalmente per il Commodore 64. Per le qualità inedite che GEOS è in grado di offrire e per lo sforzo che la realiz-

zazione di un prodotto così ben riuscito lascia supporre, auspico una volta ancora che GEOS non sia fatto oggetto di speculazioni commerciali illecite.

Booting GEOS

Il dischetto originale di GEOS presenta in directory al primo posto il file «Geos». Non è in formato GEOS e occupa un blocco del disco. La Tavola 1 mostra i primi 78 byte (esclusi i Link-byte e gli Address-byte) del file. Gli Address-byte del file indicano al C-64, se il file è caricato con «LOAD "geos", 8, 1», ove locarlo in memoria. Tale indirizzo è $0110. Nel caso, però, che l'utente carichi il file omettendo «,1», cioè come normalmente avviene per i file Basic, il programma si loca automaticamente a $0801 ove cioè ha inizio la memoria disponibile per i programmi Basic. Affinché tutto possa funzionare ancora, la prima parte del file è una linea Basic che permette di ricaricare il programma tenendo conto della locazione in memoria secondo gli Address-byte. Si tratta quindi di un file in grado di esser caricato in entrambi i modi.

Oltre ai codici listati nella Tavola 1, il file, mentre si carica nella ram del C-64, arriva a scrivere in memoria sino alla locazione $0205. Più precisamente dalla locazione $0203 alla $0205 scrive i byte #$4C #$23 #$01, che per il 6502 indicano l'istruzione «JMP $0123». Osservando attentamente sarà facile notare che questa istruzione di salto diretto cede il controllo al programma contenuto nel file «Geos» listato nella Tavola 1. Affinché si effettui l'auto-run alla fine del caricamento, l'istruzione «JMP $0123» è inserita nel Basic Input Buffer. Il piccolo programma in L.M. ha inizio col settare i colori del bordo e dello sfondo dello schermo. Per fare in modo che durante il caricamento di «Geos Boot» non compaiano i messaggi che il Kernal invia all'utente per indicare lo stato dell'operazione in corso («searching», «loading»), il programma chiama la primitive $FF90 con il registro A=# $00. Informa successivamente il Kernal delle caratteristiche del nome del file da caricare settando il registro A per la lunghezza, i registri X e Y per indicare l'indirizzo in memoria dove è memorizzato secondo la consueta norma per le routine del Kernal: X=byte-basso, Y=byte-alto. Questi dati vengono trasferiti tramite la primitive $FFBD. Il codice compreso fra $013B e $0143 si può tranquillamente omettere senza minimamente influenzare il funzionamento del programma. L'operazione finale consiste nel caricare il registro A con il codice del comando «LOAD» ed X ed Y con il valore # $FF per indicare al Kernal che il programma deve esser locato in memoria secondo gli Address-byte. La primitive $FFD5 effettua il caricamento del programma richiesto e restituisce il carry azzerato segnalando la correttezza delle operazioni. In caso contrario il programma cede il controllo alla routine Warm-Start del Basic. Se il carry è restituito azzerato, il controllo è ceduto alla locazione $6000 tramite un salto diretto. In sostanza, un piccolo programma che effettua l'auto-run in modo «elegante», offrendo anche la possibilità di essere caricato come un programma Basic e successivamente lanciato con il comando «RUN».

Affrontiamo ora l'analisi del programma «Geos Boot» che si loca a $6000 e termina a $642B. Osservando la Tavola 2, emerge chiaramente che il controllo è ancora ceduto all'indirizzo $606C. Presente inoltre una piccola routine, che ho ritenuto superfluo listare, che esegue l'operazione logica «EOR» con il valore # $27 della parte di codice che inizia a $6089 (appena dopo questa routine) e termina a $6488 (oltrepassando cioè la fine del programma). Questa parte di codice da «eorare» è importante e non ha senso se non viene trasformata. Non è molto facile quindi interpretarla su disco in quanto necessita di un'ulteriore trasformazione. Il motivo di questa trasformazione è tentare di proteggere i codici su disco da un eventuale tentativo di comprensione; si tratta comunque di un metodo molto semplice la cui realizzazione non presenta alcun ostacolo. Dopo aver operato questa trasformazione per rendere comprensibile il codice, il controllo è ceduto, tramite un salto diretto, alla locazione $60F6, ove ha veramente inizio il programma. Se volete esaminare questi codici una volta trasformati, è necessario che carichiate voi stessi in memoria il file «Geos Boot», modificare la locazione $6086 in # $60 e dare il comando, sotto monitor, «g 6000».

Se è poi vostra intenzione, provate arditamente ad effettuare il lancio del programma alla locazione $60F6, ove realmente inizia: il sistema si blocca accendendo esclusivamente il led rosso del drive. Avrebbe dovuto teoricamente funzionare, ma esiste un motivo preciso che spiega il mancato funzionamento e che verrà a suo tempo illustrato.

La Tavola 3 lista i codici dalla locazione $60F6 sino alla $6221. Questa zona del programma è la più importante in quanto è da questo punto che vengono lanciate le altre routine. Come potete osservare vi sono delle parti di codice che ho indicato come «inutili». Prima di arrivare a questa conclusione è occorso molto tempo, e questo per molteplici ragioni. Prima fra tutte il fatto che questi codici «ingannevoli» sono realmente preposti a qualcosa, per esempio inizializzano il clock del C-64 per un conteggio monostabile, settano dei nuovi vettori di interrupt oppure creano dei loop di attesa indipendenti o vincolati alla posizione del pennello elettronico sullo schermo e così via. Svolgono effettivamente delle mansioni, ma sono completamente inutili. Il grosso problema, e concedetemi una parentesi perché la questione in questo caso si complica, è consistito nel

tentativo di comprendere realmente le funzioni dei codici, e, pur operando piccole variazioni il risultato ottenuto era il medesimo. GEOS veniva caricato così come anche «Desk Top», ma non veniva abilitato il mouse! È facile capire che così stando le cose, ingannato dall'apparente intelligenza che tali codici sembravano contenere (clock, timer, conteggi, raster, interrupt...), altrettanto facile era scoraggiarsi ed assumere questi codici come indispensabili.

Il sospetto che «Geos Kernal» potesse operare un controllo sui codici di «Geos Boot», era consistente, ma lo scartai, perché in primo luogo avevo operato la variazione alla locazione $6086 e tutto procedeva per il meglio, e poi se così fosse stato, perché continuava a caricarsi sino alla fine e non abilitava il mouse, quando semplicemente si poteva resettare il sistema? Ebbi modo però di effettuare una piccola modifica alla stringa «booting geos» che appare all'inizio del booting e scoprire che il sistema si caricava completamente, ma che il mouse non veniva abilitato. Il sospetto riprendeva forma di fronte a tale risultato e continuai a indagare per capire il motivo reale. Ma dovevo evidentemente spingermi più a fondo e così feci, lasciandomi alle spalle delle zone d'ombra ancora da chiarire. Oltre queste zone «inutili» del codice di «Geos Boot» che hanno ostacolato la mia analisi, avevo incontrato un altro problema all'origine, che non mi permetteva di proseguire. Come ho già avuto modo di spiegare, «Geos Boot» a livello logico, una volta trasformato, dovrebbe essere lanciabile direttamente dalla locazione $60F6 in quanto il file «Geos» non fa altro che caricarlo in memoria. Ma operando in questa maniera si accendeva il led del drive e nulla più di significativo accadeva. Prova, ritenta, rileggi e pensa, mi è venuto in mente che forse le routine trasferite nella ram del drive controllavano la rotazione del disco per continuare il loro lavoro correttamente. Successivamente questa deduzione si rivelò errata, ma ugualmente mi permise di lanciare il programma direttamente da $60F6 preoccupandomi di porre preventivamente in rotazione il disco. Un sistema semplice, dal momento che utilizzo la cartuccia Fastload che prevede il comando «@» per comunicare col drive senza fastidiose procedure come «open», consiste nell'inizializzare il drive con il comando «I» e velocemente, prima che il disco smetta di girare, lanciare il programma con la «SYS 24822» ($60F6). Chi non possedesse la cartuccia può battere direttamente su una stessa linea la serie di comandi:

```
«OPEN 15,8,15:PRINT# 15,«I»:CLOSE15:SYS24822»
```

ottenendo il medesimo effetto. Questo era in definitiva il sistema corretto affinché tutto funzionasse lo stesso evitando il file «Geos» e la trasformazione dei codici. Queste «disavventure» si sono rivelate una notevole perdita di tempo, ma per fortuna, una volta risolte, la situazione è notevolmente migliorata. Certo è che i problemi volutamente inseriti nei codici per scoraggiare l'utente stavano sortendo l'effetto voluto. Chiudo questa parentesi per riprendere il discorso alla locazione $60F6, non spiegando i codici «inutili», perché se è stato faticoso e inutile per me...

L'istruzione «SEI» disabilita gli interrupt di sistema affinché non interferiscano con il trasferimento dei dati. Vengono poi azzerate eventuali animazioni e i bit delle maschere per gli interrupt del VIC, del CIA1 e del CIA2. Alla locazione $612C il registro direzione dati della porta A del CIA2 viene caricato con il valore # $3F (valore per altro di default all'accensione del C-64). Cerchiamo di capire in che cosa consista e come funzioni la porta A del CIA2, prima di illustrare il registro direzione dati associato. La porta A del CIA2 è un byte da 8 bit e si trova alla locazione $DD00. Il bit 0 e il bit 1, come potete osservare nella Tavola 4, sono utilizzati per cambiare il banco di memoria che il VIC sta considerando. Il bit 2 è utilizzato per il trasferimento dei dati con le periferiche secondo il protocollo RS-232. Il bit 3 trova utilizzo nella richiesta di attenzione dalle periferiche. Questi 4 bit non hanno nessuna importanza ai fini dell'analisi del trasferimento dei dati in ambiente GEOS, infatti li si troverà sempre, assumendoli come nibble, settati a # $7. Il nibble basso invece (quello di sinistra per intenderci) è di fondamentale importanza. I bit 4 e 6 sono collegati ad una linea fisica elettrica che arriva al pin 4 del connettore del bus seriale sul retro del C-64. I bit 5 e 7 sono collegati al pin 5 dello stesso connettore. Quindi essi rappresentano dei collegamenti col «mondo esterno». Per ognuno di tali bit il bus seriale necessita di una precisazione nel caso si tratti di una linea di input o di output per il C-64. Il modo per comunicare lo stato di tali bit è settare (linea di output) o azzerare (linea di input) i bit posizionalmente uguali del registro direzione dati porta A del CIA2 locato nella locazione $DD02. Potete quindi rendervi conto che alla locazione $612C tale registro è aggiornato per indicare lo stato dei bit della locazione $DD00. Per esempio, volendo settare alta la linea di clock è sufficiente «orare» la locazione $DD00 con il valore # $10. Sarà poi il cavo di collegamento del drive con il C-64 a trasportare lo stato del bus seriale alla porta presente nel drive. La lentezza nel trasferimento da imputare al drive è da considerarsi un ritardo del protocollo di trasmissione seriale (un bit singolarmente costituente un treno di bit). Una trasmissione parallela dei byte offrirebbe una velocità di trasmissione molto più elevata (i vari Speed Dos per il 1541 lavorano in questo modo).

Prima di proseguire nella trattazione di «Geos» vorrei ricordare quali sono le caratteristiche principali delle istruzioni logiche «AND», «ORA» e «EOR». Supponiamo l'esistenza di un registro A

contenente un valore qualsiasi. L'istruzione «AND» serve ad azzerare particolari bit mantenendo inalterati gli altri; l'istruzione «AND # $7F» azzera per esempio il bit 7 del registro A e lascia inalterati gli altri. L'istruzione «ORA» è utilizzata per settare particolari bit e lasciare inalterati gli altri; «ORA # $80» setta a sua volta il bit 7 tralasciando gli altri. L'istruzione «EOR» è preposta al confronto di due valori, il registro A e l'operando dell'istruzione, settando poi nel registro A i bit che dal confronto sono risultati diversi; per esempio se A=# $31 e l'istruzione è «EOR # $32», il risultato sarà A=# $03. Uno degli utilizzi più frequenti di questa istruzione è «EOR # $FF» che complementa a 2, cioè muta di segno, il valore contenuto in A. Mi sono soffermato su questi operatori logici in quanto avremo spesso a che fare con essi nel corso della trattazione.

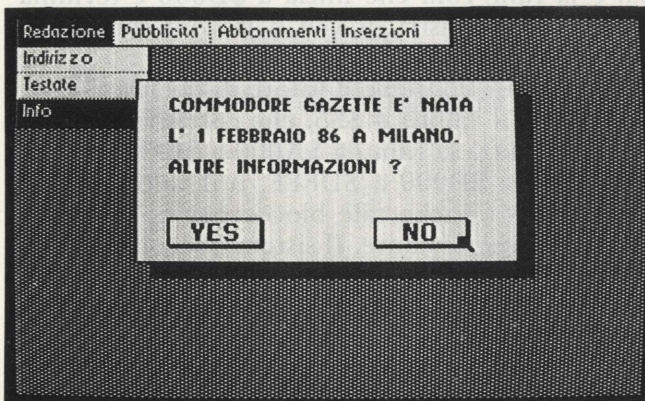
All'indirizzo $6149 e $614D vengono inizializzate due locazioni di memoria che saranno utilizzate per sincronizzare il C-64 con il drive. Il valore contenuto nella locazione $10, se trasferito nella porta A, setta a 1 i bit serial-out e clock-out. Il valore contenuto nelle locazioni $0F li resetta.

Dalla locazione $614F alla $6176 è presente una routine che pulisce lo schermo scrivendo nel contempo la stringa «booting geos». settandone opportunamente anche il colore. Sia la pulizia dello schermo che la stampa della stringa potevano, a dire il vero, essere realizzate più semplicemente utilizzando le routine del Kernal che già svolgono queste mansioni. Dalla locazione $6177 alla $61BC risiede la routine che trasferisce il turbo nella memoria del drive. Essa trasferisce il codice che inizia a $623F e termina a $642B nella ram del drive a partire dalla locazione $0300. Dal momento che il comando «M-W» high-byte lowbyte è in grado di operare solo con # $20 byte alla volta, questa routine spedisce il codice del turbo a blocchi da # $20 byte. In totale trasferisce # $10 blocchi. A $61B5 viene spedito al drive il comando «M-E $0375», richiedendo al drive di cedere il controllo all'indirizzo $0375 del turbo locato nella sua RAM. L'istruzione successiva «JMP $FFAE» dispone il bus seriale a non ricevente per terminare la trasmissione seriale.

A questo punto il drive e il computer iniziano una serie di sincronismi per procedere in coordinazione. Alla locazione $61C1 il C-64 pone la linea clock-out alta, segnale atteso, come vedremo, dal drive alla locazione $037F. Successivamente attende che il drive a sua volta risponda al segnale di sincronismo ponendo la linea serial-in bassa. Osserviamo attentamente in che modo il programma esegue l'operazione di attesa. Il ciclo di attesa è costituito dalla assai poco nota istruzione «BIT». L'istruzione «BIT» non fa altro che simulare un'operazione logica «AND» fra l'accumulatore e l'operando della stessa e settare opportunamente i bit negative, overflow e zero del Processor Status Word (PSW), operazione appunto simulata proprio perché non interessa l'accumulatore. Il bit 7 del risultato ottenuto è trasferito nel bit negative, il bit 6 del risultato relativo nel bit overflow e se il risultato è # $00 viene settato il bit zero. Tutte queste operazioni appaiono indubbiamente complicate, ma con esempi pratici la situazione può essere chiarita. Nel caso in questione, dal momento che il brunch è condizionato dal bit negative del PSW, non ha importanza il valore contenuto in A e il ciclo continua a interrogare $DD00 sino a quando non diventa positivo (bit 7 = 0), intendendo positivo anche lo # $00. Se fosse un «BPL $61C4» il ciclo sarebbe efficace sino al momento in cui $DD00 non diventa negativo (bit 7 = 1). Non avrebbe senso un ciclo di questo genere se al posto di $DD00 ci fosse una qualsiasi locazione della memoria ram che normalmente non è interessata da cambiamenti di stato indipendenti dal C-64, ma $DD00 è collegata alla porta del drive e quindi il ciclo non attende altro se non che il bit 7 si azzeri. Se invece vogliamo utilizzare l'istruzione «BIT» per analizzare lo stato di bit che non siano il bit 7 di una locazione di memoria, il ciclo che dobbiamo adottare è leggermente differente. Il brunch deve essere condizionato dal bit zero del PSW e quindi può essere «BEQ» o «BNE». Ecco un esempio:

```
$1000:  A9  04       LDA  # $04
$1002:  2C  00   DD  BIT  $DD00
$1005:  D0  FB       BNE  $1002
```

L'accumulatore è di fondamentale importanza perché indica quale bit si vuole analizzare: in questo caso il bit 2 di $DD00. Il ciclo termina quando il bit 2 diventa zero. È importante notare che il brunch non necessita di saltare a $1000 in quanto A non è alterato dall'istruzione «BIT». Il sistema migliore per imparare ad usare l'istruzione «BIT» è quello di provare realmente con dei piccoli cicli di prova. Anche se all'apparenza compli-

Un'applicazione che utilizza i menu e i box di dialogo

cata, è l'istruzione più indicata per effettuare controlli sui bit di un byte nel minor tempo possibile.

Raggiunta l'intesa fra drive e computer, entrambi iniziano a disporsi alla trasmissione. Per il momento è sufficiente osservare ciò che avviene all'interno del Commodore 64; in seguito si passerà ad analizzare il drive.

Il C-64 pone nel puntatore ($04) l'indirizzo $9000 al quale dovrà iniziare a trasferire il primo record di «Geos Kernal». Per chi non avesse seguito la prima parte della trattazione di GEOS, il file «Geos Kernal» è composto da due record, secondo la struttura VLIR, locabili a partire da due locazioni diverse nella memoria del calcolatore. La routine di $6099 si preoccupa di caricare in memoria a partire dall'indirizzo ($04) il primo record proveniente dal file; di questa ci si occuperà tra breve. Successivamente il programma controlla che anche il record sia stato caricato e a questo scopo verifica che il puntatore contenuto in ($04) sia diverso da $9000. Più precisamente opera sul byte-alto verificando che sia diverso dal valore originario #$90. Nel caso in cui sia immutato, avrà la funzione di indicare che il disco nel drive non è originale GEOS, ovvero manca dei dovuti errori memorizzati dalla Berkeley Softworks nella traccia 36. Il programma si serve allora della routine di cui ho fatto precedentemente menzione per trasformare i codici, cambiando l'istruzione «EOR # $27» in «LDA $27» e l'istruzione «JMP $60F6» in «JMP $FCE2». In questo modo occupa tutto lo spazio disponibile di codice da $6089 a $6488 con il valore # $27 e resetta il computer. Se invece il disco è originale questa routine per resettare il sistema viene ignorata e l'analisi prosegue direttamente dalla locazione $61F2.

Il puntatore ($04) viene quindi nuovamente settato perché possa contenere l'indirizzo $BF40 al quale ha inizio il caricamento del secondo record di «Geos Kernal». Il primo record occupa da # $9000 a $9FFF ed il secondo da $BF40 a $FFFF.

La routine che memorizza il record a partire dalla locazione $BF40 fino a $FFFF deve selezionare le ram sottostanti l'interprete Basic, il Kernal e la ram/rom che inizia a $D000 e termina a $DFFF (i registri di I/0 o la rom carattere). Osserveremo infatti che la locazione $01, che normalmente contiene il valore di default # $37, conterrà o il valore # $35, che corrisponde a selezionare tutta la ram da $A000 a $FFFF tranne il blocco da $D000 a $DFFF, o il valore # $30 che disalloca anche tale blocco per portare alla luce la ram sottostante. Il sistema, secondo le esigenze delle sue routine, utilizzerà o meno i codici memorizzati da $D000 a $DFFF.

Una volta caricato anche il secondo record dal file, tutti i codici di «Geos Kernal» sono stati correttamente allocati nella memoria del computer. A questo punto l'ultima operazione che il programma deve eseguire è quella di cedere il controllo alla «Cold-Start» routine di «Geos Kernal» che si trova a $C003.

Prima di addentrarmi nell'analisi delle sottoroutine menzionate, vorrei sottolineare la notevole riduzione del codice della routine principale non considerando i codici siglati come «inutili»; di fatto la routine vista nella sua struttura essenziale, limitata ai codici utili, è molto più semplice di quanto possa apparire e si accorcia inoltre notevolmente. Non è ancora il momento per iniziare a operare delle varianti in questa parte di codice, dal momento che, e la trattazione lo conferma, «Geos Kernal» effettua a sua volta un controllo sui codici.

Affrontiamo ora la sotto-routine locata a $6099. Ad essa dobbiamo inviare nel puntatore ($04) l'indirizzo al quale si desidera locare il record prelevato dal drive. Osserviamola nella Tavola 5. La routine inizia aggiornando il puntatore ($02) con l'indirizzo $623F. A questo indirizzo in cui attualmente risiedono ancora i codici del turbo, non è preposta alcuna funzione, essendo già caricato nel drive. Questa zona adesso verrà utilizzata come buffer temporaneo da 254 byte per la ricezione dei blocchi dal drive. Vi anticipo che su questo buffer «Geos Kernal» non opera nessun controllo, per ovvie ragioni. Successivamente chiama la routine $6013 dal drive memorizzando nel registro Y il numero di caratteri che desidera ricevere. Il trasferimento di un record avviene a blocchi, ovvero a settori del disco. Considerato però che l'ultimo settore del record, come in qualunque altro file, non necessariamente è composto da 254 caratteri, la trasmissione di ogni blocco è preceduta dalla trasmissione del numero di byte di cui questo è composto. Per questa ragione inizialmente, e analogamente per ogni blocco da ricevere, la routine $6013 è chiamata con Y=1. Dal momento che essa restituisce all'accumulatore l'ultimo byte ricevuto, oltre a memorizzare anch'esso nel buffer, in questo caso restituisce l'unico byte ricevuto. Esso è in definitiva il numero di byte che il drive si sta preparando a trasmetttere per il blocco in arrivo. Se tale quantità equivale a zero, ciò indica che è terminata la trasmissione del record corrente; in altre parole non ci sono più blocchi da trasmettere. Se il turbo non riconosce il dischetto come originale, trasmette subito uno zero e la routine $6099 lo interpreta come fine trasmissione del record. Dal momento che il turbo esegue il controllo di originalità alla traccia 36 prima di trasmettere qualunque blocco, nel caso di un disco copiato non viene trasmesso nessun blocco, facendo si che il puntatore ($04) punti ancora all'indirizzo $9000, con le ben note conseguenze. Se tale numero non indica invece la fine della trasmissione, viene memorizzato provvisoriamente nello stack e viene nuovamente chiamata la routine $6099 con

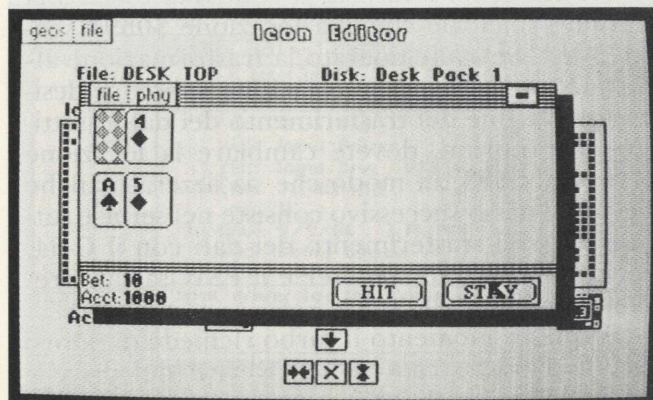
Y=numero di byte del blocco in arrivo. Il blocco, al ritorno da questa routine, si trova provvisoriamente nel buffer puntato da ($02). A questo punto il blocco deve essere trasferito nella zona di memoria prevista per il record in questione, facendo riferimento al puntatore ($04). Per effettuare questo trasferimento la configurazione della memoria del C-64 deve essere configurata «tutta ram» settando la locazione $01 della pagina zero con il valore # $30. I codici dal $60B4 al $60BF eseguono il trasferimento e ripristinano la locazione $01 con il valore $36. Prelevando dallo stack il numero di caratteri ricevuti, il programma, da $60C1 a $60CB, aggiorna il puntatore ($04) sommando a questo il numero dei caratteri. Una particolarità di questo aggiornamento è che se ($04) oltrepassa il valore $FFFF, la ricezione dei dati termina automaticamente. Terminata l'operazione di ricevimento del record, la routine cede di nuovo il controllo del programma alla routine principale.

Prima di addentrarmi nell'analisi dell'argomento più delicato (il turbo e il reale trasferimento dei dati), conviene che il nostro interesse venga spostato verso il codice trasferito nel drive, trattando successivamente il problema della comunicazione con il C-64.

Nella Tavola 6 è listata la parte di codice del turbo che contiene anche la routine principale. Essa ha inizio all'indirizzo $0375 e termine all'indirizzo $03BA. Questa contiene dei codici che ho siglato inutili, opportunamente inseriti per ostacolare l'analisi. Il comando «M-E» impartito dal C-64 per lanciare il turbo si riferisce all'inizio di questa routine. La prima operazione svolta è quella di salvare nello stack il corrente PSW e lo stack pointer locato in pagina zero all'indirizzo $49. Oltre a questa funzione il codice disabilita anche gli interrupt di sistema. Prima di continuare nell'analisi è opportuno aprire una parentesi riguardo la porta di comunicazione utilizzata dal drive per comunicare con la porta $DD00 del C-64. Essa si trova all'indirizzo del drive $1800. Si tratta di un byte da 8 bit e di questi documento solo i primi 4, vale a dire dal bit 0 al bit 3. Nella Tavola 4 potete osservare i diversi significati di ognuno di essi. Questi bit sono collegati alla linea che trasferisce i dati al C-64, collegata in ultimo ai bit della porta $DD00. Mentre però lo stato dei bit della porta $DD00 riproduce fedelmente lo stato della linea sia in input che in output, per quanto riguarda la porta del drive $1800 ci troviamo di fronte ad un comportamento leggermente diverso. In sostanza quando il drive legge lo stato delle linee interrogando i bit della locazione $1800 settati come input, lo stato della linea viene fedelmente riprodotto, analoga operazione viene effettuata nei confronti della porta $DD00 del C-64, un bit settato dal C-64 corrisponde a un bit settato della porta $1800 del drive. Quando invece il drive desidera impostare lo stato di una linea operando sui bit settati come output della locazione $1800, deve settarli per mezzo dell'operazione logica «not» che inverte lo stato binario, in questo caso, del bit. In altre parole, se si vuole settare a «1» il bit serial-in della porta $DD00, il drive deve settare a «0» il corrispondente bit serial-out della porta $1800, e viceversa. Questo diverso comportamento della porta $1800 in input e output dalla porta $DD00 è uno dei problemi da risolvere per quanto riguarda la trasmissione dei dati. Ad esso si aggiunge il fatto che, come è facile osservare, la sequenza dei bit nelle due porte è diversa. Mentre nella porta $DD00 si trovano in sequenza serial-in clock-in serial-out clock-out, nella porta $1800 la sequenza è costituita da serial-in serial-out clock-in clock-out. Dal momento che, ed è questa la principale caratteristica di potenza del turbo, come vedremo, un byte viene trasferito a coppie di due bit paralleli, per un totale di quattro invii di dati per ogni byte, la reciproca non sequenzialità dei corrispondenti bit nelle due porte costituisce un altro problema. Assumete quanto detto come un'anticipazione; in breve tutto risulterà molto più chiaro.

Per quanto riguarda il diverso comportamento della porta $1800 del drive, i dati trasferiti al C-64 saranno sempre correlati allo stesso dei bit della porta $DD00 (in ogni caso il discorso è relativo), salvo quando sarà esplicitamente dichiarato il contrario. A mo' di esempio, all'indirizzo $037A il drive «salta» alla routine $0362 che pone il bit serial-out della porta $1800 alto; di fatto pone la linea ad esso allacciato nello stesso basso e il corrispondente bit serial-in dela porta $DD00 del C-64 a «0». Assumerò sempre come dato di fatto in un'operazione di questo genere il porre a «0» il bit serial-in della locazione $DD00 nel C-64.

Riprendendo la nostra analisi alla locazione $037A il drive pone a zero il bit serial-in della porta $DD00. Questo stato sarà poi atteso dal

Il Desk Accessory Black Jack richiamato durante l'utilizzo di un'applicazione

C-64 all'indirizzo $61C4. Successivamente l'elaboratore attende che il bit clock-out della porta $DD00 sia posto a «1» dal C-64 all'indirizzo $61C1. A questo punto può apparire quanto meno strano che il C-64, dopo aver comunicato quest'ultimo bit a «1», si ponga in attesa dell'azzeramento del bit serial-in ad opera del drive, quando questa operazione è già stata svolta. Ricevuto tale conferma dal C-64, il drive accende il led rosso «orando» il bit 3 della locazione $1C00 in modo da settarlo. Questa locazione oltre a governare lo stato del led, controlla anche l'accensione e lo spegnimento del motore che mette in rotazione il disco. Ho già accennato che originariamente, cioè prima di analizzare il codice nel drive, ritenevo che il drive operasse un controllo per verificare se il motore funzionasse o meno. Di fatto non si tratta di controllo vero e proprio. Dal momento che al termine di ogni operazione col disco, il motore fa girare ancora per qualche secondo il disco stesso, nel caso in cui il turbo inizia a svolgere le operazioni di lettura del disco prima che smetta di girare, tutto procede per il meglio; se altrimenti il disco è già fermo, il turbo non si preoccupa di metterlo in rotazione di nuovo. Dal momento che normalmente il disco è ancora in rotazione a causa del caricamento di «Geos Boot» operato da «Geos», tutto funziona normalmente. Quando invece si carica il programma «Geos Boot» e lo si lancia a $6000, supposta ancora da operare la trasformazione, il led si accende, ma il disco non si mette in rotazione. Qualora desiderate ovviare al disagio che comporta porre preventivamente in rotazione il disco, potete caricare il file «Geos Boot», già trasformato, su un altro disco e operare nel codice la opportuna variazione. Non dovete far altro cioè che mutare il byte contenuto nella locazione $0388 in # $0C, in modo da settare anche il bit 2 di $1C00 che mette in rotazione il disco. È un'operazione «sicura» nei confronti del controllo operato da «Geos Kernal» perché, come ho già ricordato, questa zona di memoria non viene analizzata.

A questo punto la routine principale cede il controllo alla routine $043A per procedere con la trasmissione dei record. Effettuata l'operazione per entrambi i record, il controllo torna alla routine principale che alla locazione $0394 spegne il led. Se avete operato la trasformazione affinché il turbo ponga in rotazione il disco, e desiderate alla fine del trasferimento dei dati disattivare il motorino, dovete cambiare la locazione $0393 in #$F3, in modo che sia azzerato anche il bit 2. Il passo successivo consiste nel settare tutte le linee di trasferimento dei dati con il C-64, ripristinare lo stack-pointer e il PSW salvati precedentemente nello stack.

Da questo momento il turbo richiede di nuovo il controllo alle routine del Kernal del drive e non verrà più utilizzato.

Approfondiamo ora ulteriormente le modalità per mezzo delle quali il turbo accede ai dati del disco e li trasforma per renderli utilizzabili dal C-64.

Iniziamo quindi dalla locazione $0485 listata nella Tavola 7. Le locazioni $04EE e $04ED sono rispettivamente aggiornate con il numero del settore e il numero della traccia che la routine $03AB carica nel buffer a $0600. Si tratta infatti del settore # $01 della directory (traccia # $12), in cui risiedono i primi 8 File-entry corrispondenti alla prima pagina, secondo la suddivisione a pagine della directory che ho già trattato nel precedente articolo. Il file «Geos Kernal» non viene identificato attraverso il suo nome, ma esclusivamente per la posizione che occupa nel dischetto originale di GEOS. In altri termini il turbo, senza osservare altri elementi, preleva dal blocco caricato nel buffer a $0600 il settore e la traccia del Link-sector associato al terzo File-entry, ovvero al terzo file della prima pagina della directory. Aggiorna di nuovo le locazioni $04EE e $04ED con il nuovo settore e la nuova traccia del Link-sector del file «Geos Kernal» per poi caricarlo con la routine $03AB nel buffer prelevando i puntatori dai due record che compongono il file. Alla locazione $04A3 il turbo inizializza il puntatore ($73) al terzo byte del buffer in modo da ignorare i byte 0 e 1 che non hanno significato per il Link-sector. Il settore e la traccia ove inizia il secondo record del file sono momentaneamente salvati nello stack, mentre il settore e la traccia del primo blocco del primo record sono rispettivamente trasferiti nelle locazioni $04EE e $04ED. Tramite la routine $04C8 «Geos Kernal» preleva integralmente dal disco il primo record e lo trasmette al C-64. Aggiorna quindi $04EE e $04ED con il settore e la traccia associati al secondo record. A questo punto ha inizio la routine $04C8, con la particolarità che per il secondo record è utilizzata in modo diretto. Essa carica il blocco puntato da $04EE e $04ED nel buffer a $0600; trasferisce in Y il numero del successivo settore contenuto nel buffer alla locazione $0601 e lo memorizza in $04EE. Ma il valore contenuto in Y potrebbe anche essere, se il settore corrente è l'ultimo del record, il numero dei byte significativi. Per questa ragione, oltre a decrementare il registro Y, il programma analizza anche il byte $0600 del buffer, che corrisponde alla traccia successiva, dal momento che si tratta di # $00 indica la fine del record. Aggiorna in ogni caso anche la locazione $04ED, ma se $0600 è # $00, Y diventa la lunghezza del blocco da trasmettere, altrimenti Y viene aggiornato con la lunghezza normale di un blocco, cioè # $FE byte. Il programma arriva quindi alla locazione $04DC con Y settato a # $FE, se il blocco da spedire non è l'ultimo del record, o con Y aggiornato al numero di byte utili del settore, se il blocco da spedire è l'ultimo del record. La routine $0313 trasmette automati-

camente sia la lunghezza del blocco da spedire, sia il blocco stesso. Successivamente la routine stessa analizza $04ED: se è diverso da # $00 contiene il numero della successiva traccia a cui accedere per mezzo del blocco ritornando quindi a $04C8, se è # $00 indica che è stato trasmesso l'ultimo blocco del record e, dopo aver accordato il sincronismo con il drive con la routine $0368 (la si analizzerà in seguito), chiama la routine $0313, partendo da $0327, con A settato a # $00 per indicare la fine della trasmissione e Y settato a # $00 in quanto saltando a $0327 scavalca la decrementazione del numero di byte da spedire. Questa chiamata alla routine $0327 viene effettuata per mezzo dell'istruzione «JMP» e quindi il ritorno da subroutine della routine $0313 costituirà il ritorno da subroutine della routine listata nella Tavola 7.

Abbiamo con quest'ultima parte, portato l'analisi dei codici memorizzati nel drive allo stesso livello dell'analisi svolta sul file «Geos Boot». Da questo momento in poi, sia la routine $6013 di «Geos Boot» che la routine $0313 si occupano dell'operazione finale di trasferimento dei dati, interfacciandosi a vicenda e realizzando la veloce trasmissione che caratterizza il turbo di GEOS. In queste due routine diventa importantissima la temporizzazione, che, essendo ridotta al minimo indispensabile per poter eliminare al massimo i controlli sulla trasmissione guadagnando così tempo limite, si basa completamente sul tempo che il processore impiega per completare un ciclo macchina. A questo argomento ho intenzione di dedicare un paragrafo a sé a causa della complessità e dell'importanza che riveste per il sistema operativo GEOS.

Il turbo di GEOS

Come ho già avuto modo di accennare, il turbo di GEOS trasferisce i byte suddividendoli in coppie di due bit e trasmettendoli quindi parallelamente, utilizzando le due linee serial-out e clock-out della porta $1800 del drive. Il byte è suddiviso in due nibble, che più precisamente definirò per convenzione nibble-sinistro e nibble-destro (un nibble rappresenta la metà di un byte, cioè 4 bit). Il problema della non reciproca sequenzialità dei bit nelle due porte, fa si che una delle due interfacce trasformi opportunamente il nibble da trasmettere, oppure da ricevere. Nel protocollo di questo turbo il primo nibble, cioè il nibble-destro è trasmesso dal drive non trasformato ed è quindi il C-64 a codificarlo affinché ritorni alla configurazione del byte originale, mentre il secondo nibble, il nibble sinistro, viene trasmesso dal drive già codificato senza che a questa operazione debba assolvere il C-64. Prima di proseguire nella trattazione, analizziamo questa trasformazione per capirne il motivo.

Un nibble viene trasmesso in due fasi, utilizzando sempre le due linee serial-out e clock-out della porta $1800. Queste due linee non sono però contigue, ma separate da un bit, mentre nella porta $DD00 sono contigue. Quindi, se dal drive sono trasmessi parallelamente prima il bit 1 e 3, in quanto non sono contigui, il C-64 riceve i bit 1 e 3 contigui. Successivamente, entrambi i dati, quello pervenuto e il rimanente da spedire, sono «shiftati», il primo di due bit a destra e il secondo di un bit a sinistra, al fine di creare lo spazio necessario alla seconda parte del nibble, trasmesso con la stessa procedura. Operate le due letture, il drive si trova a gestire il nibble trasmesso con i bit da riordinare. Se il nibble originale era composto dalla serie:

b3 b2 b1 b0

il C-64 riceve la serie:

b0 b2 b1 b3

Dovrebbe, a rigor di logica, apparire evidente la necessità di trasformarlo per riottenere il nibble originale. Se però il drive opera in anticipo la trasformazione del nibble da trasmettere, facendolo diventare:

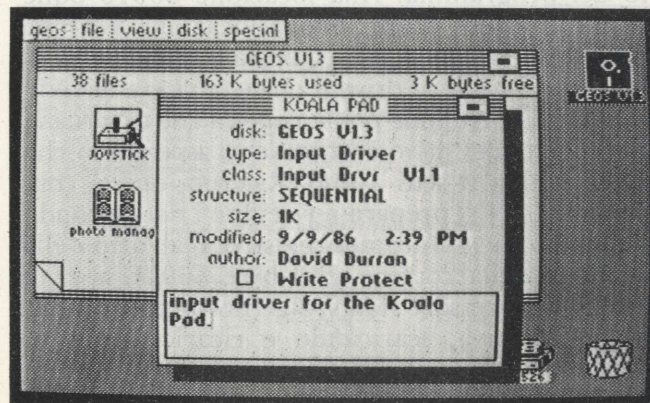
b0 b2 b1 b3

il C-64 riceve la serie:

b3 b2 b1 b0

di cui non è più necessaria la traduzione.

La trasformazione quindi può essere eseguita da una qualunque delle due interfacce. Di fatto, il primo nibble è trasmesso in forma tale da richiedere la traduzione, mentre il secondo si presenta già tradotto, così da dimezzare i tempi necessari. In questa traduzione, sia eseguita dall'una o dall'altra interfaccia, bisogna anche tener conto che il drive, scrivendo nella porta $1800, scrive i dati in negativo, di fatto invertendoli. La Tavola

INFO aumenta le informazioni associabili ad un file

8 mostra come avviene la traduzione. Si noti che le due colonne sono perfettamente interscambiabili. Il nibble da tradurre, non importa quale macchina debba tradurlo, viene utilizzato come puntatore lungo la tabella da $6003 a $6012 per il C-64 e da $0300 a $030F per il drive. Il valore puntato costituisce il nibble tradotto.

Questo modo di procedere può apparire indubbiamente complicato, ma al momento della verifica nei codici delle due interfacce dovrebbe risultare più chiaro.

Per quanto riguarda la temporizzazione, l'interfaccia del C-64, cioè la routine in questione, si comporta da «master», mentre l'interfaccia del drive si comporta da «slave». È il C-64 quindi a doversi preoccupare di temporizzare la trasmissione. Nei normali interfacciamenti seriali, la trasmissione è temporizzata da una linea di clock indipendente, in modo da conferire sicurezza per la salvaguardia dei dati, a scapito però della velocità. Nel turbo di GEOS, per iniziare la trasmissione, il C-64 richiede attenzione all'interfaccia nel drive e attende che questa venga accordata, dopodiché, però, non attende nessuna ulteriore conferma e inizia la recezione dei byte, con le modalità già illustrate lanciando un sincronismo per ognuno. Di fatto è un sincronismo piuttosto insolito, perché non attende nessuna risposta dal drive, assumendo fino a prova contraria che tutto si stia svolgendo correttamente. Lanciato tale sincronismo, una volta ricevuto l'ok dal ciclo di temporizzazione, il C-64 accede secondo una successione temporale predefinita e indipendente alla locazione $DD00, presupponendo che il drive l'abbia aggiornata utilizzando la stessa successione temporale predefinita e indipendente, in modo da procedere in coordinazione. Ho definito la successione temporale di accesso alla locazione $DD00 indipendente in quanto non verifica che il drive riesca a procedere in maniera sincronica con l'elaboratore, e lo stesso vale per il drive nei confronti del C-64.

Verifichiamo da vicino quanto detto analizzando la routine nel C-64.

All'indirizzo $6013 questa disabilita gli interrupt; si tratta di un'operazione ripetuta, ma evidentemente definita indipendentemente dal resto del programma. Alla locazione $6016 pone clock-out a zero utilizzando la locazione $0F che contiene il valore # $07. Questa è la richiesta di attenzione inviata al drive il quale la riceve all'indirizzo $036A. Attende poi la risposta che accorda la disponibilità del drive a procedere, aspettando che il bit serial-in si setti a uno. Ricevuta questa conferma il C-64 si prepara a ricevere i dati secondo il numero specificato nel registro Y all'atto della chiamata della routine. Dal byte $601E al byte $6023 il processore non svolge nessuna operazione significativa, scaricando e ricaricando solamente lo stack 3 volte; queste 6 istruzioni servono a creare un'attesa di 21 cicli macchina, corrispondente al tempo che il drive impiega per prepararsi a trasmettere. Il numero di caratteri da ricevere è memorizzato nella locazione $0E di pagina zero. I codici dalla locazione $6026 alla $6031 indicano la routine che temporizza la ricezione di ogni byte. Essa è stata volutamente complicata, ma, una volta ridotta all'essenziale, è sostituibile con la routine semplificata a lato del listato. Una rapida analisi ci permette di capire che la temporizzazione avviene analizzando la posizione del pennello elettronico sullo schermo tramite il registro Raster nella locazione $D012. Esso non fa altro che aggiornarsi continuamente sulla posizione del pennello elettronico, raggruppando più di una scansione per ogni incremento (conta da # $00 a # $FF riprendendo poi da # $00). La Tavola 12 mostra gli incrementi del nibble destro del Raster e gli stati della routine di temporizzazione associati. Essa prevede due cicli raster di attesa per ogni 6 di recezione. Questa routine è fondamentale e non può essere modificata oltre la semplificazione che ho sperimentato personalmente. Ricevuto l'ok della temporizzazione, il C-64 lancia il sincronismo (quello «insolito» per intenderci) al drive, utilizzando il contenuto della locazione $10 che è # $37. Esso setta clock-out ad uno e questo segnale è atteso dal drive all'indirizzo $0337. Lanciato il sincronismo, il C-64 lascia un tempo di 6 cicli macchina al drive per prelevare lo stato corrente della linea, per poi resettarla ed azzerare sia serial-out che clock-out. Decrementa inoltre il contatore del numero di byte da trasmettere, $0E e, dopo 6 cicli di attesa realizzati con 3 istruzioni «NOP», accede alla locazione $DD00. In totale, dal momento in cui ha lanciato il sincronismo «insolito» a quando accede a $DD00, trascorrono 21 cicli. Aumentando tale periodo di tempo le interfacce perdono il sincronismo. Da una più attenta analisi ho determinato che tale tempo è la massima attesa concessa al C-64 per prelevare i 2 bit paralleli dal drive, prima che questo aggiorni la linea con il dato successivo. I tecnici della Berkeley hanno voluto far lavorare il turbo ai limiti delle possibilità; capita infatti, anche se la versione booting di GEOS è quella originale, che durante il booting di GEOS qualche volta il turbo perda il sincronismo con il drive. Questo perché la temporizzazione lavora al limite. Se volete migliorare quindi lo sfasamento fra le temporizzazioni delle due interfacce, potete trasformare il byte $6037 dal valore # $A5 al valore # $69, in modo da trasformare l'istruzione inutile «LDA $02» da 3 cicli nell'istruzione «ADC # $02» che prevede 2 cicli macchina. Con questo sistema si può guadagnare 1 ciclo macchina. Se invece volete guadagnare 3 cicli, è opportuno mantenere inalterata la locazione $6037 e trasformare i tre «NOP» dalla locazione $6040 alla $6042, che occupano 6 cicli, nell'istruzione «JMP $6043» da 3 cicli che occupa ancora tre byte e si compone dei valori # $4C,

$43, # $60. Non provate però a guadagnare 4 cicli o più, perché incorrereste in una eccessiva anticipazione della temporizzazione da parte del C-64. Fra questi metodi è certamente consigliato il primo, in quanto opera verso il centro dello spazio temporale concesso dal drive per leggere il dato. La lettura del dato avviene leggendo la porta $DD00 e trasferendone il contenuto nell'accumulatore. Di tutto il byte però i bit che costituiscono i due bit del nibble in arrivo sono il 7 e il 6, cioè serial-in e clock-in. Dopo averli ricevuti, il C-64 li trasferisce di due posizioni a destra in modo da liberare lo spazio per gli altri due da prelevare. Dopo altri 2 cicli di attesa accede nuovamente alla porta $DD00 per completare il nibble, ma questa volta utilizzando l'istruzione «ORA» in modo da prelevare i secondi due bit senza modificare il resto del byte. Il registro A contiene in questa fase l'intero nibble ricevuto e viene spostato completamente a destra, cioè di quattro posizioni. Un particolare di cui mi stavo dimenticando è sottolineare che l'istruzione «LSR», oltre a «shiftare» di uno il byte a destra, azzera il bit 7 diventato disponibile. Il nibble ricevuto è il nibble-destro del dato finale e necessita di traduzione. Prima ancora di memorizzarlo nel registro X, il C-64 preleva in Y il nuovo stato di $DD00 e ripete la procedura in esame per prelevare i secondi due bit del secondo nibble. A questo punto A contiene il nibble-sinistro a sinistra, e con l'istruzione all'indirizzo $605A viene azzerato lo spazio per il nibble-destro. Esso, momentaneamente memorizzato in X, deve essere tradotto, per potere occupare il nibble destro di A. Per tradurlo, il suo valore è utilizzato per puntare la traduzione stessa lungo la serie di valori che inizia a $6003 e termina a $6012. In questo modo il valore puntato dal primo nibble ricevuto rappresenta la sua traduzione e viene memorizzato, tramite l'istruzione a $605C, nel registro A. È ovvio che il secondo nibble ricevuto, il nibble-sinistro, non essendo stato tradotto, è stato spedito dal drive in una forma già tradotta.

A questo punto il byte è stato completamente ricevuto ed è memorizzato nel buffer ($02) puntato da Y, preventivamente aggiornato con la locazione $0E. Se Y è diverso da zero, ciò significa che il blocco non è stato ricevuto per intero e che quindi il controllo è costretto a ritornare all'indirizzo $6026. Il C-64 termina altrimenti la recezione del blocco settando a uno la linea clock-out e restituendo il controllo alla routine di livello superiore.

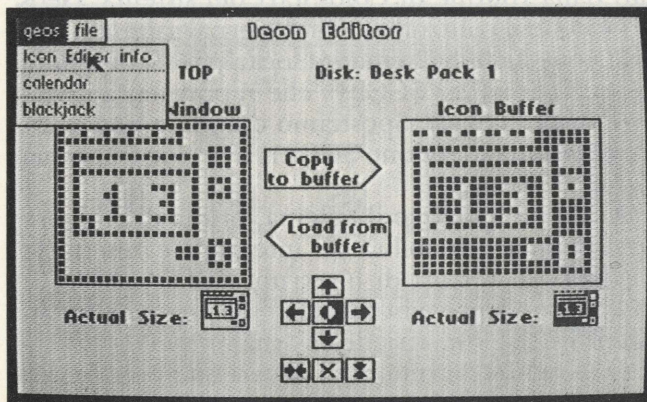
Come avete certamente avuto modo di osservare, la routine non è caratterizzata da una grande lunghezza, pur racchiudendo molte informazioni vitali per la corretta trasmissione dei dati. Quanto detto esaurisce completamente l'analisi del file «Geos Boot» e della sua interfaccia di trasmissione. Per quanto riguarda l'analisi dell'interfaccia residente nel drive, essa non è strutturata in maniera molto diversa da quella del C-64. Una delle differenze fondamentali è che questa trasmette automaticamente sia il numero di caratteri di cui è composto il blocco in arrivo, che il blocco stesso, mentre l'operazione di chiamata nel C-64 deve essere ripetuta due volte: una prima volta per ricevere il numero di caratteri del blocco in arrivo, una seconda per prelevare il blocco stesso. Nella Tavola 6 è listata l'interfaccia locata nel drive e documentata successivamente. Non è necessario soffermarci ad illustrarla nei dettagli perché è molto simile a quella appena analizzata, almeno nelle linee fondamentali, e inoltre si offre a coloro che desiderano approfondire ulteriormente l'analisi, la possibilità di completarla autonomamente.

Le due interfacce sono analoghe a quelle che verranno utilizzate dal turbo finale per la comunicazione ordinaria di GEOS con il drive, e rappresentano un valido esempio di come si può realizzare un turbo veramente veloce per il trasferimento dei dati.

GEOS Cold-Start

Per ciò che riguarda «Geos Kernal», il metodo d'analisi adottato rispecchia esattamente quello intrapreso per conoscere a fondo i segreti del C-64. Del resto la situazione era abbastanza simile: nessun manuale a disposizione.

Si tratta di conoscere il punto di inizio della routine di Cold-Start (ovvero partenza a freddo) e cercare di capire gradualmente come il sistema si inizializza. È un metodo laborioso e richiede molta pazienza, specialmente in assenza di manuali, ma porta i suoi frutti dal momento che ci si può notevolmente avvicinare alla struttura portante di «GEOS». GEOS, trattandosi di un sistema operativo, necessita anch'esso di una routine di Cold-Start, atta a svincolare completamente la precedente organizzazione della memoria riorganizzandola secondo il nuovo standard. Ci sono però delle locazioni della memoria il cui significa-

L'applicazione Icon Editor permette di personalizzare le icone associate ai file

to non può essere cambiato in quanto sono assegnate dalla Commodore a particolari mansioni per il controllo dell'I/0, della grafica del suono e della memoria. Esse non saranno quindi riutilizzate per altri scopi.

Si è visto nel precedente paragrafo che Geos Kernal si divide in due record locati in zone differenti di memoria. Il primo inizia a $9000 e termina a $9FFF, il secondo rispettivamente a $BF40 e termina a $FFFF. Queste due aree di memoria contengono qualche locazione che il sistema necessita trasformare frequentemente anche se almeno momentaneamente possono essere assunte come aree di rom, trattandosi in realtà di memoria ram. GEOS, oltre a utilizzare le pagine 0, 1, 2, 3 della memoria ram per la memorizzazione dei dati di sistema, considera lo spazio da $8000 a $8BFF come un ulteriore ampliamento della memoria disponibile per la memorizzazione dei dati di sistema. Se a questo si aggiunge che da $8C00 a $8FE7 risiede la pagina di colore della bit map (1000 caratteri in totale), e da $6000 a $7F3F la bit map in ombra (8000 caratteri; ci riserviamo di trattarla in seguito), risulta che lo spazio disponibile per le applicazioni utente inizia a $0400 e termina a $5FFF. All'apparenza può sembrare molto limitativo, ma un buon programma subordinato a GEOS deve poter utilizzare tutte le routine già implementate nel sistema operativo, e conseguentemente i suoi codici risultano molto più semplici e concisi. Quindi anche se lo spazio disponibile non è abbondantissimo, le potenzialità di GEOS permettono ugualmente di considerarlo più che sufficiente, in particolar modo per i programmi in L.M. In questo modo sono strutturati i due applicativi geoPaint e geoWrite. Non sarebbe peraltro possibile realizzare programmi di tale complessità con soli 23 kbyte (23552 byte, in esadecimale $5C00) disponibili per l'utente. In ogni caso GEOS prevede la possibilità di salvare provvisoriamente delle zone dei propri codici su disco per aumentare lo spazio disponibile all'utente. Questa qualità è per esempio utilizzata per i Desk Accesory, anche se i codici salvati su disco appartengono all'applicazione corrente, e non a Geos Kernal.

Nella Tavola 9 è listata una parte dei codici utilizzati dalla routine di Cold-Start del sistema. Geos Boot cede il controllo a Geos Kernal saltando alla locazione $C003. In questa locazione è contenuta l'istruzione «JMP $CC21» che rappresenta il vettore di Cold-Start. Spostiamo quindi la nostra attenzione alla locazione $CC21, ove cioè il sistema è inizializzato.

A questo indirizzo il sistema, prima di svolgere altri compiti, disabilita gli interrupt e resetta lo stack. I tre vettori di interrupt $FFFA, $FFFC, $FFFE sono provvisoriamente settati con l'indirizzo $C2E4 ove risiede l'istruzione «RTI»; successivamente saranno aggiornati anche loro. La vera e propria inizializzazione si attua con la routine $CBF5, anch'essa presente nel listato. È certo una delle più importanti per l'avvio corretto di GEOS. Essa inizia azzerando il flag decimalmode del PSW e saltando alla routine $CD6B. Questa routine aggiorna i registri di I/0 utilizzando il vettore indicizzato che si trova a $CF83. La struttura di questo vettore, e la relativa routine $CE73 che lo riloca, è molto interessante per aggiornare zone diverse di memoria in una sola volta. Chiamando la routine $CE73, è necessario aggiornare il puntatore ($02) con l'indirizzo di inizio del vettore. Esso è suddiviso in gruppi differenziati, ognuno a rilocamento indipendente sequenziale. Ogni gruppo di dati è locabile in maniera indipendente dagli altri nella zona di memoria assegnata dal vettore. I primi due byte del vettore rappresentano anche i primi due byte del primo gruppo e indicano l'indirizzo al quale tale gruppo di dati deve sequenzialmente locarsi.

Il terzo byte indica il numero di byte che compone il gruppo. Grazie a queste informazioni, la routine $CE73 è in grado di locare il primo gruppo all'indirizzo specificato e identificare l'inizio del gruppo successivo lungo il vettore. Il numero di gruppi di cui può essere composto il vettore è arbitrario. Per segnalare la fine del vettore alla routine $CE73, i due byte successivi all'ultimo gruppo devono essere entrambi posti a zero. Per osservare ciò praticamente è possibile analizzare sia la routine $CE73 che la $CD6B che la utilizza. Nella Tavola 10 è listato il programma Resume GEOS in L.M. Sarà analizzato successivamente, anche se è possibile immediatamente constatare che utilizza la routine $CE73 per rilocare un vettore. Tramite quindi la routine $CE73, $CD6B inizializza gli I/0 utilizzando il vettore locato a $CF83. Successivamente chiama la routine $CEC8 settando il puntatore ($02) con l'indirizzo $CD4D, al quale corrisponde un vettore esclusivamente sequenziale, e il registro Y con la relativa lunghezza. $CEC8 non ha altra funzione che quella di locare l'intero vettore a partire dall'indirizzo $D000, per la lunghezza specificata dal registro Y. Essa è quindi una routine espressamente dedicata all'aggiornamento dei registri del chip grafico. Un particolare di questa routine è rappresentato dal fatto che, se lungo il vettore sequenziale un byte, o più d'uno, ha un valore di # $AA, il corrispondente byte nel CHIP grafico viene lasciato inalterato. Gli aggiornamenti più salienti sono costituiti dalla scelta della pagina grafica della bit map, locandola a $A000, e l'apparizione del mouse, che però in questa fase attualmente si trova ancora fuori campo. Analizzando tale vettore sarete in grado di modificare le dimensioni del mouse in altezza e larghezza ($D017 e $D01D), renderlo multicolor e altre particolarità. La routine successiva chiamata da $CD6B è la $CDE1. Essa deve essere chiamata dopo aver selezionato la rom del Kernal del C-64 in via provvisoria, in maniera da permetterle di

copiare la normale serie di indirizzi nella zona da $0314 a $0334 dedicata ai vettori di interrupt. Quando tale routine cede di nuovo il controllo all'elaboratore è necessario ripristinare la ram sottostante il Kernal del C-64, per potere quindi selezionare il Kernal di GEOS. L'ultima routine chiamata da $CD6B è la $FE80. Essa corrisponde alla prima locazione della zona del Kernal dedicata alla gestione del joystick. Questa zona si estende sino alla locazione $FFF9 e corrisponde esattamente ai dati contenuti nel file «Joystick» di tipo Input-Driver residente sul dischetto originale di GEOS. Saltando alla locazione $FE80, il sistema setta i parametri di spostamento posizionali del mouse e lo stato corrente dello stesso. Questi parametri sono indicati nel paragrafo dedicato alla gestione del mouse; si tratta delle locazioni $8506, $8507, $3A, $3B, $3C. La routine $CD6B ha così termine; riassumendo, essa inizializza i registri di I/0, i registri del VIC selezionando la pagina bit map corrente e abilitando l'apparizione dello sprite 0, la serie di vettori di interrupt da $0314 a $0334 e i parametri posizionali e di comando del mouse.

Torniamo dunque alla routine $CBF5 e procediamo con la chiamata della routine $CD31. Essa, praticamente, utilizzando la rotuine $CE73, loca il vettore indicizzato che ha inizio a $CF3F. Questo vettore inizializza alcune locazioni di pagina zero e alcune della zona di memoria da $84B8. L'aggiornamento più importante è costituito dal puntatore ($31) che contiene l'indirizzo $84C1, al quale normalmente risiedono i dati dello sprite del mouse. Un particolare importante per l'analisi è rappresentato dal fatto che questo vettore può essere a discrezione modificato, diventando così di notevole utilità nella verifica dello stato dinamico della memoria, settandolo in modo tale che il sistema assuma come sprite la zona di 63 byte che si intende visionare. Una volta operata questa trasformazione, lo sprite diventa l'immagine trasparente della parte di memoria che volete osservare, offrendovi la possibilità di leggerne il contenuto mentre il sistema è in funzione. Se l'immagine dello sprite è troppo piccola è possibile ottenere una espansione in altezza e larghezza al fine di una maggiore comodità. Questo metodo di indagine un po' insolito si è dimostrato utilissimo per confermare la funzione di particolari zone di memoria. Proseguendo lungo $CBF5 incontriamo la routine $CDED. Essa svolge parecchie funzioni importanti. Inizia aggiornando $8501 con il valore # $7F. Questa locazione è la velocità massima del mouse. La locazione $8502 è la velocità minima del mouse ed è settata a # $1E. La locazione $8503 rappresenta l'accelerazione del mouse, cioè il tempo impiegato dal mouse per raggiungere le velocità massima partendo dalla minima, ed è settata con il valore # $7F. Per questi tre registri il valore che contengono deve rimanere nel range # $00-# $7F, compresi gli estremi. Proseguendo, il sistema utilizza la routine $C17B per riempire la parte di memoria corrispondente alla pagine di colore, associata alla bit map, con il relativo carattere contenuto nella locazione $851E. Questa zona si estende dall'indirizzo $8C00 al $8FE7. La routine $C17B è di assai facile utilizzo. Essa necessita che $06 contenga il tipo di carattere, $03 il numero di blocchi da 256 caratteri che si vuole riempire, $02 l'offset finale dell'ultimo blocco, che non deve essere necessariamente composto da 256 caratteri, e che il puntatore ($04) indichi l'indirizzo a partire dal quale poter procedere allo riempimento. Il carattere utilizzato per riempire la pagina del colore è # $BF; il nibble destro rappresenta il colore del fondo così come il sinistro il colore dei tratti. Questa routine si rivela molto utile nella gestione dell'alta risoluzione. Aggiorna quindi il colore delle due animazioni che potrebbero essere usate, il mouse e le icone fantasma, e memorizza lo sprite del mouse dall'indirizzo $84C1, prelevandone i dati dall'indirizzo $BF40.

Continuiamo dunque l'analisi della routine $CBF5. Il byte $848C è aggiornato con il numero del drive corrente. Le inizializzazioni successive sono indispensabili alla routine $C2B0 per l'aggiornamento della lista dei device collegati, nella quale risiedono i numeri associati per ognuno, lo stato corrente e, nel caso del drive, quale di questi è quello contenente il turbo ed il relativo stato, segnalando se quest'ultimo è da lanciare o è già stato lanciato. Il numero di device corrente è prelevato dalla locazione $BA. La routine $CFA1 aggiorna la data e l'ora del sistema. La routine $C2E5 controlla lo stato del drive corrente e lo prepara per la comunicazione. Termina a questo punto la fase più importante delle routine utilizzate da GEOS per inizializzarsi.

Riprendiamo l'analisi della routine di Cold-Start $CC21 dalla locazione $CC3E. La routine $DBAE è una tra quelle incontrate dalla più complessa analisi. Quando ho illustrato il controllo effettuato da Geos Kernal sui codici di Geos Boot, e del problema che comportavano piccole

Driver per la penna ottica, il mouse e la tavoletta grafica

variazioni insignificanti su quei codici (si caricava completamente, ma escludendo l'abilitazione del mouse), mi riferivo a questa routine. Essa controlla la memoria da $6000 a $623E, e da $6358 a $6429. Se queste due zone di memoria sono state alterate, la routine non setta il vettore $FFFE all'indirizzo $E2DC. A questo indirizzo si trova la routine di interrupt più importante del sistema, quella che si preoccupa cioè dell'aggiornamento dei timer, del movimento e del controllo del mouse e della tastiera. Se questa routine non setta opportunamente il vettore $FFFE, il sistema si carica, ma non si realizza l'interfacciamento con l'utente. Il programma Resume GEOS listato di Tavola 10 è stato realizzato al fine di modificare questa routine in modo che setti $FFFE senza effettuare il controllo sui codici di Geos Boot, che a sistema avviato sono sostituiti dalla pagina bit map in ombra. Questo piccolo programma deve essere lanciato con il comando «G 6000» da monitor, o «SYS 24576» da Basic. Esso sostituisce anche l'istruzione «JSR $FD33» con tre «NOP» in modo che la routine $FD33 non sia eseguita. Essa infatti esegue un ulteriore controllo che per riesumare GEOS risulta essere fatale. L'utilità di Resume GEOS consiste nel fatto che vi permette di caricare GEOS dal dischetto originale, di resettare il computer per mezzo appunto dell'apposito tasto, oppure di uscire da GEOS con l'opzione «BASIC», e riesumare completamente il programma. Una volta lanciata per la prima volta, il sistema può essere resettato a propria discrezione e riattivato con un «g CC21» sotto monitor. Ogni volta però, compresa la prima, è necessario spegnere e riattivare il drive.

In questo modo sarete in grado di effettuare a vostra volta una diretta analisi dei codici di GEOS, operando qualche variazione e osservandone i risultati. A questo punto, per l'analisi, diventa importante il monitor residente nella cartuccia Fastload, capace di lasciarvi osservare tutta la ram sottostante il Kernal, la zona da $D000 a $DFFF e l'interprete. Il programma Resume GEOS è, nella sua semplicità (ma quale fatica arrivarci...), un potente asso nella manica per iniziare a conoscere GEOS ed eventualmente diventare dei programmatori in standard GEOS.

Abbiamo tralasciato la routine $CDA3. Questa è definibile come il tasto «CLR» del C-64 atto a ripulire lo schermo. Analizzandola, viene subito messo in evidenza che essa scrive alternati i byte # $55 (locazione $CDB8) e # $AA (locazione $CDBF) nelle due pagine bit map che GEOS normalmente gestisce. I due caratteri generano praticamente il reticolo di fondo caratteristico dello scenario offerto da GEOS. Cambiandoli è possibile cambiare lo sfondo. Bisogna fare attenzione al fatto che Desk Top utilizza un'altra routine di cancellazione dello schermo che permette cancellazioni parziali, e quindi, anche se avete cambiato lo sfondo, Desk Top lascerà intatta la parte superiore dello schermo, che tra l'altro mai lo interessa, e resetterà la parte inferiore secondo il proprio disegno di sfondo. $CDA3 azzera due pagine bit map. La più importante inizia a $A000 e termina a $BF3F. Essa contiene la schermata normalmente visibile. L'altra inizia a $6000 e termina a $7F3F. Questa a sua volta rappresenta la schermata in ombra, ovvero il luogo dove sono memorizzate le informazioni sottostanti le finestre aperte, in modo da essere riscritte contemporaneamente alla chiusura della finestra. Un comportamento di questo genere si nota effettivamente all'apertura delle finestre e dei menu.

La routine successiva è la $CDED che abbiamo già avuto modo di incontrare. A questo punto GEOS setta la locazione $848D con il valore # $01 per il numero di drive collegati al C-64 e la locazione $8465 con il valore # $00. A questa locazione normalmente risiede il nome del drive di stampa prescelto. Dal momento che è Desk Top a selezionare il drive di default (cioè il primo che incontra analizzando la directory), GEOS aggiorna, a questo livello, la zona dedicata al nome con una stringa nulla. Eccoci giunti all'indirizzo $CC54. Questa locazione rappresenta l'inizio della routine di Warm-Start (partenza a caldo), alla quale le applicazioni cedono il controllo per ricaricare Desk Top. Il programma finestra listato nella Tavola 11 salta a questa locazione per ritornare quindi a Desk Top. Da questa locazione alla $CC61 sono ripetute le routine più importanti dell'inizializzazione. Da questo momento in avanti ci preoccuperemo di analizzare i codici preposti al caricamento ed al lancio di Desk Top.

GEOS e il drive

Analizziamo a questo punto le routine che permettono di accedere al disco caricando un programma, o i singoli settori di questo. Il turbo caricato nel drive opera con modalità diverse da quelle illustrate precedentemente. Questo turbo è sostanzialmente in grado di ricevere comandi da GEOS e di svolgere contemporaneamente operazioni molto diverse. I comandi sono trasmessi da GEOS inviando al drive l'indirizzo della routine che deve eseguire, ed eventualmente dei parametri dipendenti dal tipo di comando. Per quanto riguarda il caricamento dei record, è GEOS stesso che deve caricare settore per settore dal disco per mezzo di un apposito comando relativo ai vari settori. Di fatto, il drive 1541 diventa un'unità subordinata al programma stesso contravvenendo alle prerogative che lo caratterizzavano al suo debutto come strumento indipendente, in quanto è GEOS a guidarne i processi di accesso al disco.

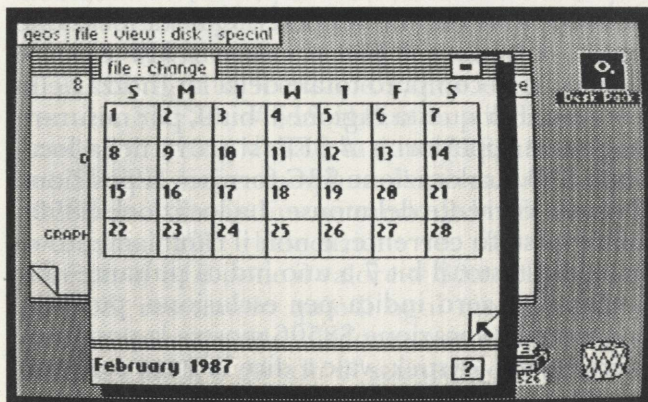
Iniziamo la nostra analisi scorrendo la main routine $CC21; ci troviamo quindi all'indirizzo $CC62. La routine $CCD2 aggiorna il puntatore ($0E) con l'indirizzo al quale si trova il nome del

file sul quale vogliamo eseguire delle operazioni. Questa routine prepara ($0E) a puntare alla stringa «Desk Top» che inizia a $CCDB. La routine $C20B ricerca il nome del file scorrendo la directory pagina per pagina. Essa restituisce due flag per documentare lo stato dell'operazione. Come tutte le routine di I/0, è in grado di restituire X a zero per indicare la corretta procedura nel trasferimento delle informazioni con il disco. X diverso da zero indica invece un errore commesso dal drive. Il registro Y segnala inoltre il successo dell'operazione. Se viene azzerato significa che il file, scorrendo la directory, è stato trovato. Se Y vale # $05, significa che il file non è presente sul disco, ma il disco è formattato GEOS. Se Y vale # $FF, il file non è stato trovato e il disco non è formattato GEOS. Se l'operazione ha avuto successo, cioè se il file è stato trovato in una pagina della directory (compreso il Desk-sector), il settore della directory ove si trova è memorizzato da $8000 a $80FF. Nel caso in cui il sistema debba accedere al Desk-sector, il settore # $00 della traccia # $12 viene caricato da $8200 a $82FF. L'indirizzo $8000 è restituito nel puntatore ($0A). Nel puntatore ($0C) si trova l'indirizzo da $8000 a $80FF a partire dal quale ha inizio il File-entry del file in questione. $04 e $05 contengono la traccia/settore memorizzata a $8000. Come ultima fase, l'intero File-entry del file è memorizzato grazie alle successive elaborazioni da $8400 a $841D. Questa routine, $C20B, utilizza tutta una serie di routine per accedere al drive. La prima che si può incontrare ed analizzare è la routine $C9F7 che carica il settore # $01 della traccia # $12 a $8000. Restituisce in $04/$05 la traccia/settore caricati a $8000, il puntatore ($0C) aggiornato all'indirizzo $8002 (questo valore permette a ($0C) di puntare l'inizio del primo File-entry) e il valore # $00 nella locazione $886E. La routine $C461 carica invece la traccia/settore desiderata, memorizzata in $04/$05 a $8000. La routine $C469 carica quindi il t/s puntato da $04/$05 nella ram identificata dal puntatore ($0A). La differenza fra le due consiste nel fatto che la seconda permette di memorizzare nella ram il settore prescelto nel punto desiderato. Per scorrere i File-entry lungo il settore memorizzato a ($0A), il puntatore ($0C), che punta all'inizio di ogni File-entry, deve essere di volta in volta aggiornato per essere in grado di puntare al successivo. A questo proposito la routine $CA10 somma a ($0C) il valore # $20; nel caso però che, scorrendo il settore memorizzato a ($0A), si oltrepassi l'ultimo File-entry presente nel settore, la routine carica il blocco successivo della directory a ($0A). Nel caso in cui la directory abbia termine, la routine carica il Desk-sector a ($0A). In entrambi i casi la routine resetta il puntatore ai File-entry ($0C). La routine $CA5A controlla, caricando il settore # $00 della traccia # $12 a $8200, se il disco è formattato GEOS. Il controllo avviene unicamente per la prima parte della targhetta («GEOS FORMAT»), contenuta all'indirizzo $CBB2. Non viene controllata tutta la targhetta offrendo così la possibilità alla versione corrente di GEOS di considerare GEOS compatibili anche dischetti formattati da altre versioni di GEOS. Y è restituito azzerato unicamente nel caso in cui il disco sia formattato GEOS, e in $04 e $05 si trovano la t/s del Desk-sector.

Tornando alla main routine $CC21, alla locazione $CC69 avviene il controllo sugli errori di trasmissione. Se si verifica un errore di trasmissione imputabile a GEOS, di qualunque natura esso sia, tramite la locazione $848D, controlla se i drive collegati sono più di uno. In tal caso inverte il numero di device considerato per provare l'accesso al secondo drive, trasformandolo da # $08 a # $09 o viceversa, e chiamando la routine $C2B0 che possa così aggiornare le tabelle dei device collegati. Nel caso invece il file non sia stato trovato sul disco corrente e non sia collegato un secondo drive il sistema salta alla gestione della finestra di dialogo che richiede di inserire un disco contenente Desk Top. Se invece il file è stato trovato, il sistema può caricarlo attraverso la routine $C208. Essa differenzia il caricamento in base al tipo di file, secondo la tabella illustrata nell'articolo precedente, selezionando la routine da eseguire. Caricato il file le istruzioni dalla $CCA5 alla $CCCB si preoccupano di lanciarlo. Se volete procedere ad una prova pratica di queste operazioni, potete sostituire il nome «Desk Top» con il nome «geoPaint» e noterete che rilanciando GEOS sarà caricato e lanciato «geoPaint».

GEOS e il mouse

La Berkeley ha dotato GEOS di un'interfaccia molto sofisticata adibita al controllo del mouse. Essa prevede infatti il computo dell'accelerazione del mouse. Questo controllo si rivela più intelligente di quanto si pensi. Avrete notato, settando l'accelerazione a un valore basso tramite Preference Mgr, che il mouse, impartito il comando,

Il Desk Accessory Calendar, un'agenda computerizzata

impiega un certo tempo a raggiungere la velocità massima. Tenendo poi il mouse fermo per poche frazioni di secondo e ricominciando quindi a muoverlo, noterete che esso non riparte con la velocità minima di spostamento, ma da un valore superiore. Mantenendolo inattivo per più tempo vi accorgerete che la velocità di spostamento iniziale diminuisce ulteriormente; se per un periodo più lungo sarà tenuto inattivo, il mouse si muoverà con la minima velocità selezionata. Questo indica che il mouse-controller è in grado di capire da quanto tempo il mouse si trova in stato di inattività rispetto all'ultimo comando impartito eseguito. La locazione $8507 indica il contatore dell'accelerazione. Esso ha il compito di iniziare il conteggio dal valore della velocità minima impostata alla velocità massima. Inizia infatti a contare quando il mouse comincia a muoversi. L'accelerazione imposta il tempo che tale contatore impiega per portarsi al valore della velocità massima. Ad ogni suo incremento, la velocità del mouse aumenta. Il contatore incrementa il valore conteggiato sin tanto che il mouse continua ad essere in movimento. Nel momento in cui il mouse viene reso inattivo, la locazione $8507 inizia a decrementarsi con la velocità impostata dal valore dell'accelerazione. Se poi il movimento del mouse riprende, il contatore $8507 non riprende il computo dal valore minimo di velocità, ma dal valore cui era giunto decrementandosi. Una gestione così raffinata dell'accelerazione è una qualità, peraltro ottima, tra quelle che caratterizzano GEOS. Per settare la velocità massima, la minima e l'accelerazione, si agisce sulle locazioni rispettivamente $8501, $8502, $8503; il valore massimo è # $7F e il valore minimo è rappresentato da # $00 (vi consiglio di non uscire da questo range). La locazione $39 permette di verificare il cambiamento di stato nella direzione del mouse (compresa la direzione «nulla», o mouse fermo) e la pressione del tasto. Il bit 6 passa per un breve tempo a uno se il mouse cambia stato (cioè nel caso in cui la leva del joystick si muove); il bit 5 passa per un breve periodo di tempo a uno se cambia lo stato dello sparo. Attenzione al fatto che per stato viene inteso lo stato corrente. La locazione $3A fornisce la posizione orizzontale corrente del mouse sullo schermo, calcolandola secondo la larghezza del pixel. Questa locazione misura dal valore # $00 al valore # $FF e non è sufficiente al computo totale della larghezza dello schermo. Per questa ragione il bit 9, per misurare lunghezze superiori a # $FF, si trova nella locazione $3B. La locazione $3C fornisce la posizione verticale corrente del mouse. La locazione $8505 indica lo stato corrente, e non il cambiamento di stato, del tasto; il bit 7 a uno indica pulsante non premuto, a zero indica per esclusione, pulsante premuto. La locazione $8506 mostra la posizione corrente del joystick, vale a dire lo stato corrente del mouse, e non, al contrario, il cambio di stato. I valori che essa può assumere e i movimenti associati sono listati nella Tavola 12. Quando il mouse «clicka» dentro una finestra di un dialogo, all'interno di uno dei minibox («OK», «CANCEL», «OPEN», ecc.), il numero corrispondente per identificare il tipo di risposta si trova alla locazione $851D e può variare dal valore # $01 al valore # $06. I minibox saranno oggetto di trattazione del paragrafo seguente. È possibile anche stabilire l'area di spostamento del mouse sullo schermo, in modo da condizionarne i limiti di azione. L'area di spostamento deve avere la forma di un quadrato e per definirla sono sufficienti quattro lati. L'angolo a sinistra in alto dello schermo è considerato il limite minimo di spostamento del mouse. Normalmente il mouse è libero di muoversi su tutta la superficie dello schermo; affinché ciò avvenga è necessario che le locazioni dalla $84B8 alla $84BD siano aggiornate come segue. La locazione $84B8 vale # $00 e indica l'altezza, dal lato superiore dello schermo, del limite orizzontale superiore di spostamento. La locazione $94B9 ha il valore di # $C7 e misura l'altezza, dal lato superiore dello schermo, del limite orizzontale inferiore di spostamento. La locazione $84BA è uguale a # $00 e segnala la posizione, rispetto al lato verticale sinistro dello schermo, del limite verticale sinistro di spostamento; la locazione $84BB vale # $00 ed è il nono bit associato. La locazione $84BC corrisponde a # $3F e mostra la posizione, rispetto al lato verticale sinistro dello schermo, del limite verticale destro di spostamento; la locazione $84BD vale infine # $01 ed è il nono bit associato. La routine $EB7E setta queste locazioni affinché il mouse possa muoversi su tutta la superficie dello schermo. Per concludere, la routine di interrupt locata a $E2DC, oltre a controllare il mouse, controlla anche la tastiera, ed il valore Ascii del tasto premuto si trova alla locazione $8504. Con quest'ultima notizia ritengo d'avere esaurito le nozioni di mio possesso sul mouse e sulla tastiera. Esse sono frutto di un lavoro capillare di analisi e sono più che sufficienti per permettere a chiunque di iniziare a programmare in standard GEOS. Affrontiamo adesso l'ultimo argomento che mi propongo di trattare a conclusione di questi due articoli sul mondo di GEOS.

GEOS e la grafica

In quest'ultimo paragrafo ci occupiamo di analizzare in che modo avvengono gli output di sistema. Esempio di questo è la finestra di dialogo che appare nel caso in cui GEOS debba accedere al disco per caricare Desk Top ed abbia difficoltà nel reperirlo. All'indirizzo $CC88 il puntatore ($02) è aggiornato alla locazione $CCE4, ove inizia il vettore di dati che caratterizzerà la finestra e le scritte in essa contenute. Il puntatore ($02) viene poi inviato alla routine $C256 che interpre-

ta il vettore e disegna la finestra. Analizziamo quindi il significato di ogni byte di cui è composto: il primo byte, a $CCE4, possiede due significati diversi; il nibble-sinistro può presentare il bit numero 7 settato o azzerato. Se è settato, indica che i dati delle dimensioni fisiche della finestra devono essere prelevati da quelli della finestra standard prevista da GEOS. Se azzerato indica invece che i dati fisici della finestra seguono dal byte successivo. Nel caso che stiamo trattando, le dimensioni fisiche sono quelle standard, e sono prelevate dalla locazione $F329 in poi per un totale di sei byte. Il primo byte indica l'altezza del bordo superiore dello schermo. Il byte successivo indica l'altezza del bordo inferiore riferita al lato superiore dello schermo; il terzo misura la distanza del lato sinistro dal bordo sinistro dello schermo; il byte successivo è il nono bit associato. Il quinto byte è preposto alla indicazione della distanza del lato destro della finestra dal lato sinistro dello schermo; il byte successivo è il nono bit. Se il bit 7 del primo byte del vettore è azzerato, questi dati devono seguire la prima locazione del vettore. Il programma «Finestra» listato nella Tavola 11 genera una finestra di dimensioni non standard, e i primi sei byte dopo il primo nel vettore rappresentano quindi le dimensioni fisiche della finestra. Il nibble-destro del primo byte del vettore alla locazione $CCE4 indica il tipo di carattere con il quale si desidera che sia disegnato lo sfondo della finestra. I caratteri disponibili sono gli stessi utilizzati da geoPaint per mutare lo sfondo e sono numerati da # $00 al # $22. Essendo il bit 7 settato, il byte successivo al primo non fa parte delle dimensioni fisiche della finestra. Esso vale # $0B ed indica che i dati successivi definiscono una linea di caratteri. Questo valore deve precedere i dati di ogni linea di stampa. Il terzo e il quarto byte segnalano in quale posizione dello schermo deve iniziare la linea di caratteri da stampare. Il terzo indica il margine sinistro riferito al bordo della finetra e il quarto il margine superiore riferito al bordo superiore della finestra. I due byte successivi mostrano alla routine di visualizzazione della finestra da quale indirizzo di memoria iniziare a prelevare i caratteri di cui è composta la linea di stampa. La prima linea da stampare per esempio inizia alla locazione $CCF8. Le linee da stampare possono contenere un numero arbitrario di caratteri di controllo. Il carattere di controllo è un byte che può variare fra i valori # $08 e # $1B ed è analizzato dalla routine di visualizzazione di un carattere che si trova a $DE11. Se il carattere è identificato da un codice ASCII non contenuto nel range # $08-# $1B, non viene considerato dalla routine un carattere di controllo ed è visualizzato normalmente. Alcuni caratteri di controllo necessitano di parametri e questi, nel numero determinato dal tipo di controllo, devono seguire il carattere di controllo lungo la stringa corrente. Ad ogni carattere è assegnata una routine che lo esegue. Analizziamoli sequenzialmente: # $08 è eseguito dalla routine $DF3A; # $09 dalla routine $DE9E. #$0A somma alla corrente posizione verticale del prompt per l'output del carattere il valore contenuto nella locazione $29 e il carry a 1; all'operazione è preposta la routine $DEAA # $0B azzera la posizione del prompt posizionandolo nell'angolo sinistro superiore dello schermo; lo esegue la routine $DEB2. # $0C è opposto al carattere di controllo # $0A, in quanto sottrae alla corrente posizione verticale del prompt il valore contenuto in $29 e il carry a 1; è eseguito dalla routine $DEBB. # $0D è eseguito dalla locazione $DEC3; # $0E dalla routine $DECE e setta il bit 7 a 1 del byte $2E, in modo che le scritte vengano stampate con sottolineatura. # $0F setta il bit 7 a 0 del byte $2E, in modo da disabilitare la sottolineatura delle frasi; trova esecuzione all'indirizzo $DED5. # $10 è eseguito dalla routine $DF60. # $11 non possiede alcun compito specifico dal momento che corrisponde alla routine $E2DB composta a sua volta solo dall'istruzione «RTS». # $12 setta a 1 il bit 5 della locazione $2E e abilita il reverse delle scritte; è eseguito alla locazione $DEDC. #$13 disabilita il reverse delle scritte ed è portato a termine alla locazione $DEE3. # $14 è un controllo che mette in grado l'utente di mutare completamente la posizione orizzontale del prompt sullo schermo, aggiornandola con i 2 byte successivi lungo la stringa, che indicano rispettivamente la nuova posizione orizzontale e il nono bit associato. Questo carattere di controllo necessita di due parametri ed è eseguito alla locazione $DEEA. # $15 permette di mutare completamente la posizione verticale del prompt sullo schermo, aggiornandola con il byte successivo lungo la stringa, indicante la nuova posizione verticale; a compiere questa operazione è abilitata la locazione $DF01. # $16 comprende entrambi gli ultimi due controlli e necessita quindi di tre parametri sequenziali: rispettivamente nuova posizione orizzontale, nono bit associato, nuova posizione verticale. È eseguito dalla routine $DF0E. # $17 si tratta di un carattere di controllo che ordina al puntatore, il cui compito è quello di scorrere la stringa, di saltare i 2 byte successivi, perché non siano né interpretati né visualizzati; relativa a ciò è la locazione $DF14. # $18 abilita lo stile in grassetto settando a 1 il bit 6 della locazione $2E; è eseguito dalla routine $DF20. # $19 abilita lo stile italico settando a 1 il bit 4 della locazione $2E; alla locazione $DF27 viene compiuta l'operazione. # $1A abilita lo stile outline settando il bit 3 a 1 della locazione $2E; l'esecuzione è alla locazione $DF2E. # $1B pone il valore # $00 nella locazione $2Z, resettando quindi ogni modo e stile di stampa; corrisponde alla locazione $DF35. Ricordo che i modi e gli stili di stampa possono essere sovrapposti. Tramite quindi la presenza del carat-

tere di controllo all'interno di una riga di stampa, si è in grado di modificare l'output dei caratteri seguenti in maniere diverse. La linea deve però terminare con un byte di valore # $00. Per ogni linea che si vuole visualizzare nella finestra è necessario quindi specificare, oltre al carattere # $0B che la caratterizza, altri quattro byte. Terminata la serie di dati delle linee di stampa, vengono forniti i dati dei minibox che si desidera visualizzare e controllare. Il primo byte per definire un minibox ne stabilisce il tipo secondo la seguente tabella:

# $01	OK
# $02	CANCEL
# $03	YES
# $04	NO
# $05	OPEN
# $06	DISK

I due byte successivi mostrano la posizione del minibox sullo schermo. È possibile definire più di un minibox, in modo che ne compaiano diversi per offrire all'utente diverse possibilità di scelta. Il vettore di dati termina con un byte di valore # $00. Una volta che la routine di visualizzazione $C256 ha posto in essere i minibox, attende che il mouse «clicki» su uno di questi. Scelto il minibox, il suo tipo è memorizzato nella locazione $851D e la finestra viene «chiusa» riscrivendo sulla pagina grafica la parte della schermata che aveva coperto. In questo modo la finestra può essere aperta su qualunque videata, e alla sua chiusura il disegno coperto viene ripristinato. Nel programma Finestra listato nella Tavola 11 ho creato una finestra seguendo queste specifiche. Se sostituite l'istruzione «JSR $CDA3» con tre «NOP», potrete osservare il ripristinamento della schermata coperta dalla finestra. Attenzione che per abilitare il mouse è stato assunto che sia già stato utilizzato almeno una volta il programma resume GEOS, che modifica la routine $DBAE in maniera da settare il solo puntatore ($FFFE)= $E2DC. Se desiderate invece che «Finestra» funzioni anche senza aver lanciato «Resume GEOS», modificatelo così che setti autonomamente il vettore ($FFFE)=$E2DC, abilitando il mouse, senza rendersi necessaria la chiamata della routine $DBAE.

Con quest'ultimo argomento ha quindi termine l'analisi del sistema operativo GEOS. Le basi sulle quali poggiano le cognizioni su questo programma si possono ritenere sufficienti per intraprendere possibilmente un lavoro di programmazione con questi nuovi standard operativi. GEOS è infatti una novità assoluta ed offre, a chi lo vuole, ampi spazi per mettere alla prova le proprie capacità in vista della realizzazione di software inedito e di sicuro successo, dal momento che manualistica tecnica relativa a questo pacchetto non è ancora stata pubblicata (è comunque in preparazione da parte della IHT, la casa editrice di Commodore Gazette, n.d.r.).

Gli strumenti dunque vi sono stati forniti: manca solo la vostra fantasia e la vostra creatività.

Tavola 1

```
0110  12              ???                  Piccolo programma
0111  08              PHP                  basic di una
0112  0A              ASL   A              istruzione:
0113  00              BRK                  10 LOAD «GEOS»,8,1
0114  93              ???                  nel caso GEOS sia
0115  20  22  47      JSR   $4722          caricato con locazione
0118  45  4F          EOR   $4F            nella RAM BASIC A $0801
011A  53              ???                  obbligata. Necessita di un
011B  22              ???                  RUN per proseguire
011C  2C  38  2C      BIT   $2C38
011F  31  00          AND   ($00),Y
0121  00              BRK
0122  00              BRK
0123  A9  0E          LDA   # $0E
0125  8D  20  D0      STA   $D020          colore del bordo
0128  A0  06          LDY   # $06
012A  8C  21  D0      STY   $D021          colore dello sfondo
012D  A9  00          LDA   # $00
012F  20  90  FF      JSR   $FF90          disabilita l'apparizione di messaggi del Kernal
0132  A9  09          LDA   # $09          N° CARATTERI NOME DEL FILE
0134  A2  55          LDX   # $55          IL NOME INIZIA A $0155
0136  A0  01          LDY   # $01
0138  20  BD  FF      JSR   $FFBD          SETTA PARAMETRI NOME
013B  A9  50          LDA   # $50          CODICE
013D  A6  BA          LDX   $BA            INUTILE
013F  A0  01          LDY   # $01
0141  20  BA  FF      JSR   $FFBA
0144  A9  00          LDA   # $00          COMANDO DI LOAD
0146  A2  FF          LDX   # $FF          PER INDICARE NESSUN RILOCAMENTO
0148  A0  FF          LDY   # $FF
014A  20  D5  FF      JSR   $FFD5          LOAD PROGRAMMA
014D  90  03          BCC   $0152
014F  6C  02  03      JMP   ($0302)        ERRORE, JMP BASIC WARM START
```

```
0152  4C  00  60  JMP  $6000    OK, CEDE IL CONTROLLO
0155  47          ???           NOME DEL FILE
0156  45  4F      EOR  $4F      GEOS BOOT
0158  53          ???
0159  20  42  4F  JSR  $4F42
015C  4F          ???
015D  54          ???
```

Tavola 2

```
,6000  4C  6C  60  JMP  $606C     Qui è ceduto il controllo e inizia GEOS BOOT. A606C EORA
                                  la mem.da $6089 a $6048 e effettua JMP $60F6. E una delle
                                  mini protezioni.
,6003  0F          ???            Tavola di conversione
,6004  07          ???            dei dati in ingresso
,6005  0D  05  0B  ORA  $0B05     dal Drive
,6008  03          ???
,6009  09  01      ORA  #$01      Questa è la tavola di conversione
,600B  0E  06  0C  ASL  $0C06     del NIBBLE destro della parola, che
,600E  04          ???            è trasmesso per primo, ma non tradotto.
,600F  0A          ASL
,6010  02          ???
,6011  08          PHP
,6012  00          BRK
,6013  78          SEI            Disabilita Interrupt
,6014  A5  0F      LDA  $0F       Clear serialout e clockout, serialin e clockin
,6016  8D  00  DD  STA  $DD00
,6019  2C  00  DD  BIT  $DD00     WAIT BIT7 = 1 Attende che il drive si ponga in ascolto
,601C  10  FB      BPL  $6019
,601E  48          PHA            21 cicli macchina
,601F  68          PLA            di attesa
,6020  48          PHA
,6021  68          PLA
,6022  48          PHA
,6023  68          PLA
,6024  84  0E      STY  $0E       0E = N° DI CARATTERI DA RICEVERE
,6026  38          SEC            SEC
,6027  AD  12  D0  LDA  $D012     LDA $D012
,602A  E9  31      SBC  #$31      SBC $# 01
,602C  90  04      BCC  $6032     NOP
                                  NOP
,602E  29  06      AND  #$06      AND # $06
,6030  F0  F5      BEQ  $6027     BEQ $6027
,6032  A5  10      LDA  $10       Ricevuto l'OK dalla temporizzazione
                                  trasmette OK per inizio trasmissione dati e non attende
                                  conferme
,6034  8D  00  DD  STA  $DD00     setta serial out, clock out
,6037  A5  02      LDA  $02       3 cicli macchina è il limite per il passo di recezione. Meglio 69 02
,6039  A5  0F      LDA  $0F
,603B  8D  00  DD  STA  $DD00     Clear la porta per ricevere i 2 bit alla volta
,603E  C6  0E      DEC  $0E       Decrementa il n° di caratteri da ricevere
,6040  EA          NOP            6 cicli
,6041  EA          NOP
,6042  EA          NOP
,6043  AD  00  DD  LDA  $DD00     Preleva in serialin e clockin i primi 2 bit del nibble non tradotto
                                  di destra
,6046  4A          LSR
,6047  4A          LSR            Li sposta a destra di 2 bit per lasciare posto agli altri due
,6048  EA          NOP            2 cicli
,6049  0D  00  DD  ORA  $DD00     Preleva i secondi 2 bit del nibble non tradotto di destra
,604C  4A          LSR
,604D  4A          LSR            Sposta il nibble ricevuto 4 bit a destra, cioè a inizio Byte. I Bit a
                                  sinistra,
,604E  4A          LSR            cioè il nibble sinistro, sono tutti a 0, posti da LSR
,604F  4A          LSR            Questo è il nibble non tradotto, è da tradurre.
,6050  AC  00  DD  LDY  $DD00     Preleva i primi 2 bit del nibble tradotto di sinistra
,6053  AA          TAX            X = Memorizza il nibble destro già ricevuto, che punterà la
                                  tabella di conversione
,6054  98          TYA
,6055  4A          LSR            Li sposta a destra di 2 bit per lasciare posto agli altri due
,6056  4A          LSR
,6057  0D  00  DD  ORA  $DD00     Preleva i secondi 2 bit del nibble tradotto di sinistra
,605A  29  F0      AND  #$F0      Azzera il nibble destro
,605C  1D  03  60  ORA  $6003,X   Trasferisce la traduzione del nibble destro, prelevata dalla
                                  tabella di conversione
```

```
,605F  A4  0E       LDY  $0E
,6061  91  02       STA  ($02),Y      Memorizza il byte nel buffer
,6063  D0  C1       BNE  $6026        If y> 0 get next byte
,6065  EA           NOP               2 cicli attesa
,6066  A2  17       LDX  #$17         Clockout = 1 fine recezione del blocco.
,6068  8E  00  DD   STX  $DD00
,606B  60           RTS
```

Tavola 3

```
,60F6  78           SEI               Disabilitazione degli interrupt
,60F7  A9  36       LDA  #$36         Codice
,60F9  85  01       STA  $01          Inutile
,60FB  A9  01       LDA  #$01
,60FD  8D  04  DD   STA  $DD04
,6100  A9  00       LDA  #$00
,6102  8D  1A  D0   STA  $D01A        Codice
,6105  8D  05  DD   STA  $DD05        Inutile
,6108  8D  15  D0   STA  $D015        Nessuna animazione visualizzata
,610B  A9  7F       LDA  #$7F
,610D  8D  19  D0   STA  $D019        Clear i bit della maschera interrupt register
,6110  8D  0D  DC   STA  $DC0D        Clear i bit della maschera per IRQ, interrupt control register
                                      CIA 1
,6113  8D  0D  DD   STA  $DD0D        Clear i bit della maschera per NMI, interrupt control register
                                      CIA 2
,6116  A9  62       LDA  #$62         Codice
,6118  8D  15  03   STA  $0315        Inutile
,611B  A9  39       LDA  #$39
,611D  8D  14  03   STA  $0314
,6120  A9  62       LDA  #$62
,6122  8D  19  03   STA  $0319
,6125  A9  3E       LDA  #$3E
,6127  8D  18  03   STA  $0318
,612A  A9  3F       LDA  #$3F         Setta direzione dati porta A.
,612C  8D  02  DD   STA  $DD02
,612F  A9  81       LDA  #$81
,6131  8D  0D  DD   STA  $DD0D        Codice
,6134  A9  09       LDA  #$09         Inutile
,6136  8D  0E  DD   STA  $DD0E
,6139  A0  2C       LDY  #$2C
,613B  AD  12  D0   LDA  #D012
,613E  C5  10       CMP  $10
,6140  F0  F9       BEQ  $613B
,6142  85  10       STA  $10
,6144  88           DEY
,6145  D0  F4       BNE  $613B
,6147  A9  37       LDA  #$37
,6149  85  10       STA  $10          Valore da usare in porta A: setta serialout e clockout
,614B  A9  07       LDA  #$07
,614D  85  0F       STA  $0F          Valore da usare in porta A: clear serialout e clockout
,614F  A9  04       LDA  #$04
,6151  85  07       STA  $07          CLEAR SCHERMO
,6153  A9  00       LDA  #$00
,6155  85  06       STA  $06
,6157  A2  04       LDX  #$04
,6159  A0  00       LDY  #$00
,615B  A9  20       LDA  #$20
,615D  91  06       STA  ($06),Y
,615F  C8           INY
,6160  D0  F9       BNE  $615B
,6162  E6  07       INC  $07
,6164  CA           DEX
,6165  D0  F4       BNE  $615B
,6167  A0  0F       LDY  #$0F         Plotta 16 caratteri in centro schermo
,6169  A9  0E       LDA  #$0E         definendone anche il colore nella pagina del colore
,616B  99  4C  D9   STA  $D94C,Y
,616E  B9  89  60   LDA  $6089,Y
```

```
,6171  99  4C  05   STA  $054C,Y    Booting Geos...
,6174  88           DEY
,6175  10  F2       BPL  $6169
,6177  A9  62       LDA  # $62      Trasferisce il turbo nella RAM del drive a partire da $0300
,6179  85  0B       STA  $0B
,617B  A9  3F       LDA  # $3F
,617D  85  0A       STA  $0A        (0A) = $623F inizio del turbo da caricare in drive memoria
,617F  A9  03       LDA  # $03
,6181  8D  D6  60   STA  $60D6
,6184  A9  00       LDA  # $00
,6186  8D  D5  60   STA  $60D5      (60D5) = $0300 indirizzo iniziale M-W $ nella RAM del drive
,6189  A9  10       LDA  # $10
,618B  85  0C       STA  $0C        Contatore blocchi da spedire da 32 byte ciascuno
,618D  20  D7  60   JSR  $60D7      Spedisce 32 caratteri nel drive A (60D5) indirizzo
,6190  18           CLC
,6191  A9  20       LDA  # $20      Offset da sommare a (0A) per puntare i 32 caratteri successivi
,6193  65  0A       ADC  $0A
,6195  85  0A       STA  $0A
,6197  90  02       BCC  $619B      (0A) = (0A) + OFFSET
,6199  E6  0B       INC  $0B
,619B  18           CLC
,619C  A9  20       LDA  # $20      (60D5)=(60D5) + OFFSET
,619E  6D  D5  60   ADC  $60D5
,61A1  8D  D5  60   STA  $60D5      Genera il nuovo indirizzo di M-W per puntare la RAM del drive
,61A4  90  03       BCC  $61A9      per il prossimo blocco da 32 bytes
,61A6  EE  D6  60   INC  $60D6
,61A9  C6  0C       DEC  $0C        Decrementa il contatore di blocchi da 32 byte
,61AB  D0  E0       BNE  $618D      If $0C > 00 then spedisci nuovo blocco
,61AD  A9  60       LDA  # $60
,61AF  85  03       STA  $03
,61B1  A9  CD       LDA  # $CD      (02) = 60CD (02) punta a M-E $0375
,61B3  85  02       STA  $02
,61B5  20  22  62   JSR  $6222      Spedisce M-E $0375 al drive. Lancio del turbo
,61B8  20  AE  FF   JSR  $FFAE      Dispone il bus a non ricevente
,61BB  78           SEI             Codice
,61BC  A0  21       LDY  # $21      Inutile
,61BE  88           DEY
,61BF  D0  FD       BNE  $61BE
,61C1  20  65  60   JSR  $6065      Pone clockout = 1. Atteso dal Drive a $037F
,61C4  2C  00  DD   BIT  $DD00
,61C7  30  FB       BMI  $61C4      Wait bit 7=0 serialin=0. Posto del Drive a $037A
,61C9  A9  90       LDA  # $90
,61CB  85  05       STA  $05        (04) Punta inizio RAM nel C-64 ove si locherà 1° record
,61CD  A9  00       LDA  # $00      di GEOS KERNAL
,61CF  85  04       STA  $04        (04) = $9000
,61D1  20  99  60   JSR  $6099      Lo carica e lo loca a partire da (04)
,61D4  A5  05       LDA  $05
,61D6  C9  90       CMP  # $90
,61D8  D0  12       BNE  $61EC      Il record è stato caricato. Il disco è originale
,61DA  A9  A9       LDA  # $A9
,61DC  8D  7A  60   STA  $607A      Il disco è copiato
,61DF  A9  E2       LDA  # $E2      Trasforma EOR #$27 a $607A in $607A LDA #$27
,61E1  8D  87  60   STA  $6087      Trasforma il JMP 60F6 finale in
,61E4  A9  FC       LDA  # $FC      JMP $FCE2 reset C-64
,61E6  8D  88  60   STA  $6088
,61E9  4C  6C  60   JMP  $606C      Cancella tutto il codice da $6089 a $6488 e JMP FCE2
,61EC  AD  12  D0   LDA  $D012      Codice
,61EF  8D  FE  02   STA  $02FE      Inutile
,61F2  A9  BF       LDA  # $BF
,61F4  85  05       STA  $05        (04) Punta inizio RAM nel C-64 ove si locherà 2° record
,61F6  A9  40       LDA  # $40      di GEOS KERNAL
,61F8  85  04       STA  $04        (04) = $BF40
,61FA  20  99  60   JSR  $6099      Lo carica e lo loca a partire da (04)
,61FD  AD  12  D0   LDA  $D012
,6200  F0  FB       BEQ  $61FD
,6202  8D  FF  02   STA  $02FF      Codice
,6205  A5  0F       LDA  $0F        Inutile
,6207  8D  00  DD   STA  $DD00
,620A  A9  7F       LDA  # $7F
,620C  8D  0D  DD   STA  $DD0D
,620F  AD  0D  DD   LDA  $DD0D
,6212  A5  BA       LDA  $BA
,6214  20  B1  FF   JSR  $FFB1
,6217  A9  EF       LDA  # $EF
,6219  20  93  FF   JSR  $FF93
,621C  20  AE  FF   JSR  $FFAE
,621F  4C  03  C0   JMP  $C003      Cede il controllo a GEOS KERNAL
```

Tavola 4

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
$DD00	Serial in	Clock in	Serial out	Clock out	Atn out	RS-232 out	VIC II addr15	VIC II addr14
$1800					clock out	clock in	serial out	serial in

Tavola 5

```
,6099  A9  62       LDA  #$62     (02) Punta all'inizio del buffer da 254 caratteri utilizzato per il
,609B  85  03       STA  $03      trasferimento dei blocchi DRIVE → C-64
,609D  A9  3F       LDA  #$3F
,609F  85  02       STA  $02      (02)=$623F
,60A1  A0  01       LDY  #$01     N° di caratteri da ricevere per il dato in arrivo
,60A3  20  13  60   JSR  $6013    Get il carattere. Indica il n° di caratteri del settore in arrivo
,60A6  A8           TAY           Y = n° di caratteri da ricevere per il blocco in arrivo
,60A7  F0  23       BEQ  $60CC    Ricevuto 00 come n° di caratteri del blocco in arrivo. Quindi
                                  fine trasmissione
,60A9  48           PHA           N° di caratteri da ricevere per il blocco in arrivo → Stack
,60AA  20  13  60   JSR  $6013    Riceve il n° di caratteri previsto da Y e li mette nel buffer (02)
,60AD  68           PLA           A = n° caratteri ricevuti
,60AE  48           PHA
,60AF  A8           TAY           Y = n° caratteri ricevuti
,60B0  A9  30       LDA  #$30     Setta il C-64 tutta RAM (64 K) per poter scrivere anche nella
                                  Ram da $D000 - $DFFF normalmente non usato
,60B2  85  01       STA  $01
,60B4  88           DEY           Effettua il trasferimento del buffer ricevuto (y= n° caratteri
                                  ricevuti)
,60B5  B1  02       LDA  ($02),Y  in (04), cioè nella RAM del C-64 assegnata
,60B7  91  04       STA  ($04),Y
,60B9  98           TYA
,60BA  D0  F8       BNE  $60B4
,60BC  A9  36       LDA  #$36     Ripristina
,60BE  85  01       STA  $01
,60C0  68           PLA           A = n° di caratteri ricevuti nel buffer
,60C1  18           CLC
,60C2  65  04       ADC  $04      (04) = (04) + A
,60C4  85  04       STA  $04      Se (04) da $FFFF diventa $0000
,60C6  90  D9       BCC  $60A1    esce automaticamente da questa routine
,60C8  E6  05       INC  $05
,60CA  D0  D5       BNE  $60A1    go to get next blocco
,60CC  60           RTS
```

Tavola 6

```
,0300  0F                         Tabella di conversione
,0301  07                         dei dati per la routine di posizionamento testina
,0302  0D  05  0B                 (71) = n° di byte da trasmettere
,0305  03                         Y = 1-1 già decrementato per trasmettere 1 byte = n° byte del
,0306  09  01                     blocco in arrivo
,0308  0E  06  0C                 Si prepara a trasmettere send ready to C-64
,030B  04                         A = n° di caratteri da trasmettere
,030C  0A                         Lo spedisce
,030D  02                         Y = n° di caratteri da trasmettere
,030E  08
,030F  00
,0310  20  40  60
,0313  84  71       STY  $71
,0315  A0  00       LDY  #$00

,0317  20  68  03   JSR  $0368
,031A  A5  71       LDA  $71
,031C  20  27  03   JSR  $0327
,031F  A4  71       LDY  $71
```

```
,0321  20  68  03  JSR  $0368      Si prepara a trasmettere i send ready to C-64
,0324  88          DEY             Y = Y-1
,0325  B1  73      LDA  ($73),Y    Preleva dal buffer partendo dall'alto il dato
,0327  AA          TAX             X = Dato
,0328  4A          LSR
,0329  4A          LSR             Isola a destra il NIBBLE sinistro e
,032A  4A          LSR             azzera il nibble attuale sinistro
,032B  4A          LSR
,032C  85  70      STA  $70        $70 = 0000 nibble sinistro da spedire tradotto
,032E  8A          TXA             A = Dato
,032F  29  0F      AND  # $0F      Azzera il nibble sinistro
,0331  AA          TAX             X = 0000 nibble destro spedito non tradotto
,0332  A9  04      LDA  # $04
,0334  8D  00  18  STA  $1800      Clockin = 1 (la linea clockout di C-64 azzerata)
,0337  2C  00  18  BIT  $1800      Wait clockin = 1 sincronismo inizio trasmissione
,033A  F0  FB      BEQ  $1337
,033C  2C  00  18  BIT  $1800      10 cicli macchina
,033F  D0  00      BNE  $1341
,0341  D0  00      BNE  $1343
,0343  8E  00  18  STX  $1800      Trasmette del nibble destro il 1° bit e il 3° da sinistra
,0346  8A          TXA
,0347  2A          ROL             Sposta di 1 bit a sinistra
,0348  29  0F      AND  # $0F      Azzera il nibble di sinistra
,034A  8D  00  18  STA  $1800      Trasmette del nibble destro il 2° e il 4° bit da sinistra
,034D  A6  70      LDX  $70
,034F  BD  00  03  LDA  $0300,X    Trasmette la traduzione del nibble sinistro
,0352  8D  00  18  STA  $1800      Tradotta con la tabella. Il 1° e il 3° bit da sinistra
,0355  EA          NOP             2 cicli
,0356  2A          ROL             Sposta 1 bit a sinistra
,0357  29  0F      AND  # $0F      Azzera il nibble di sinistra
,0359  C0  00      CPY  # $00      Setta PSW per il confronto di Y = n° byte ancora da spedire
,035B  8D  00  18  STA  $1800      Trasmette il 2° e il 4° bit da sinistra
,035E  D0  C4      BNE  $1324      Trasmetti il prossimo byte
,0360  F0  00      BEQ  $1362      Trasmetti fine trasmissione dei
,0362  A2  02      LDX  # $02      blocco
,0364  8E  00  18  STX  $1800      Pone serialout = 1 (clear serial in C-64)
,0367  60          RTS
,0368  A9  04      LDA  # $04
,036A  2C  00  18  BIT  $1800      Attende bit 2 = 0 clock in = 0
,036D  D0  F9      BNE  $1368
,036F  A9  00      LDA  # $00      Pone bit 1=0 serial out = 0 (setta serial in in C-64)
                                   Send ready to C-64. Attende richiesta di attenzione da C-64 e
                                   l'accorda.
,0371  8D  00  18  STA  $1800
,0374  60          RTS
,0375  08          PHP             Inizio turbo. Salva corrente PSW on Stack
,0376  78          SEI             Disabilita interrupt interni
,0377  A5  49      LDA  $49
,0379  48          PHA             Salva on stack lo stack pointer
,037A  20  62  03  JSR  $0362      Ponte serial out = 1
,037D  A9  04      LDA  # $04
,037F  2C  00  18  BIT  $1800
,0382  F0  FB      BEQ  $137F      Wait bit 2=1 clock in=1
,0384  AD  00  1C  LDA  $1C00
,0387  09  08      ORA  # $08      Accende led. Se ora # $0C accende anche il motore
,0389  8D  00  1C  STA  $1C00
,038C  20  3A  04  JSR  $043A      Routine di trasmissione file
,038F  AD  00  1C  LDA  $1C00
,0392  29  F7      AND  # $F7      Spegne led. Se AND # $F3 spegne anche il motore
,0394  8D  00  1C  STA  $1C00
,0397  20  8F  F9  JSR  $F98F      Codice inutile
,039A  A9  00      LDA  # $00
,039C  8D  00  18  STA  $1800      Setta tutte le linee di trasferimento dei dati
,039F  85  33      STA  $33        Codice
,03A1  A9  EC      LDA  # $EC      inutile
,03A3  8D  0C  1C  STA  $1C0C
,03A6  68          PLA
,03A7  85  49      STA  $49        Ripristina stack pointer
,03A9  28          PLP             Ripristina PSW
,03AA  60          RTS             Back to dos 1541
,03AB  AD  ED  04  LDA  $04ED      Posiziona la traccia contenuta in $04ED
,03AE  20  BB  03  JSR  $03BB
,03B1  20  19  04  JSR  $0419      Prova a caricare nel buffer il settore puntato da $04ED, $04EE
,03B4  A5  00      LDA  $00        Controlla il codice di stato restituito
,03B6  C9  01      CMP  # $01
,03B8  D0  F7      BNE  $13B1      Se non è status = "OK", prova a ricaricarlo.
,03BA  60          RTS
```

Tavola 7

```
,047D  A9  12       LDA  # $12      Se il disco è copiato
,047F  20  BB  03   JSR  $03BB      Posiziona testina $12 traccia e
,0482  4C  E4  04   JMP  $04E4      spedisce fine trasmissione
,0485  A9  01       LDA  # $01
,0487  8D  EE  04   STA  $04EE      04EE = Settore
,048A  A9  12       LDA  # $12
,048C  8D  ED  04   STA  $04ED      04ED = Traccia
,048F  20  AB  03   JSR  $03AB      Carica il settore nel buffer a $0600
,0492  AD  44  06   LDA  $0644
,0495  8D  EE  04   STA  $04EE      Settore del link-sector associato al 3° file (riferendosi al file entry)
,0498  AD  43  06   LDA  $0643      presente nella prima pagina della directory
,049B  8D  ED  04   STA  $04ED      Traccia come sopra
,049E  20  AB  03   JSR  $03AB      Carica il link-sector nel buffer a $0600
,04A1  A9  06       LDA  # $06
,04A3  85  74       STA  $74        Posiziona (73) che punta al 1° byte utile del link-sector,
                                    saltando il byte 0 e il byte 1 non significativi
,04A5  A9  02       LDA  # $02      (73) = $0602
,04A7  85  73       STA  $73
,04A9  AD  05  06   LDA  $0605      Settore del 1° blocco di dati del 2° record di GEOS KERNAL
,04AC  48           PHA             on stack
,04AD  AD  04  06   LDA  $0604      Traccia del 1° blocco di dati del 2° record di GEOS KERNAL
,04B0  48           PHA             on stack
,04B1  AD  03  06   LDA  $0603
,04B4  8D  EE  04   STA  $04EE      Settore del 1° blocco di dati del 1° record di GEOS KERNAL
,04B7  AD  02  06   LDA  $0602
,04BA  8D  ED  04   STA  $04ED      Traccia del 1° blocco di dati del 1° record di GEOS KERNAL
,04BD  20  C8  04   JSR  $04C8      carica e spedisce tutto il 1° record
,04C0  68           PLA
,04C1  8D  ED  04   STA  $04ED      Traccia inizio 2° record
,04C4  68           PLA
,04C5  8D  EE  04   STA  $04EE      Settore inizio 2° record
,04C8  20  AB  03   JSR  $03AB      Carica il settore nel buffer A $0600
,04CB  AC  01  06   LDY  $0601      Y = next sector se $6000=0 n° di byte ultimo settore se $6000=0
,04CE  8C  EE  04   STY  $04EE      (04EE) = next sector
,04D1  88           DEY             Se fosse il n° di byte ultimo settore del record
,04D2  AD  00  06   LDA  $0600
,04D5  8D  ED  04   STA  $04ED      (04ED) = next traccia
,04D8  F0  02       BEQ  $14DC      Se $6000=0 dobbiamo trasmettere l'ultimo settore e quindi lascia
                                    Y=invariato
,04DA  A0  FE       LDY  # $FE      Non è l'ultimo settore e quindi Y=# $FE (254)
,04DC  20  13  03   JSR  $0313      Trasmette la lunghezza del blocco in arrivo e il blocco
,04DF  AD  ED  04   LDA  $04ED      Next traccia o indicatore di fine record (se è $00)
,04E2  D0  E4       BNE  $14C8      Se non era l'ultimo settore prosegue con il prossimo
,04E4  20  68  03   JSR  $0368      Attende richiesta di attenzione da C-64 e l'accordo
,04E7  A9  00       LDA  # $00      Carattere da spedire (fine trasmissione)
,04E9  A8           TAY             Y=0 n. 1 caratteri da spedire
,04EA  4C  27  03   JMP  $0327      Spedisce il carattere di fine trasmissione
,04ED  27           ???             del record.
,04EE  D8           CLD
```

Tavola 8

Nibble ricevuto		Nibble spedito	
#$0	0 0 0 0	#$F	1 1 1 1
#$1	0 0 0 1	#$7	0 1 1 1
#$2	0 0 1 0	#$D	1 1 0 1
#$3	0 0 1 1	#$5	0 1 0 1
#$4	0 1 0 0	#$B	1 0 1 1
#$5	0 1 0 1	#$3	0 0 1 1
#$6	0 1 1 0	#$9	1 0 0 1
#$7	0 1 1 1	#$1	0 0 0 1
#$8	1 0 0 0	#$E	1 1 1 0
#$9	1 0 0 1	#$6	0 1 1 0
#$A	1 0 1 0	#$C	1 1 0 0
#$B	1 0 1 1	#$4	0 1 0 0
#$C	1 1 0 0	#$A	1 0 1 0
#$D	1 1 0 1	#$2	0 0 1 0
#$E	1 1 1 0	#$8	1 0 0 0
#$F	1 1 1 1	#$0	0 0 0 0

Tavola 9

```
,CBF5 78           SEI
,CBF6 D8           CLD
,CBF7 20  6B  CD   JSR   $CD6B
,CBFA 20  31  CD   JSR   $CD31
,CBFD 20  ED  CD   JSR   $CDED
,CC00 A9  08       LDA   # $08
,CC02 8D  8C  84   STA   $848C
,CC05 A0  03       LDY   # $03
,CC07 A9  00       LDA   # $00
,CC09 99  8E  84   STA   $848E,Y
,CC0C 88           DEY
,CC0D 10  F8       BPL   $CC07
,CC0F A4  BA       LDY   $BA
,CC11 A9  01       LDA   # $01
,CC13 99  86  84   STA   $8486,Y
,CC16 98           TYA
,CC17 20  B0  C2   JSR   $C2B0
,CC1A 20  A1  CF   JSR   $CFA1
,CC1D 20  E5  C2   JSR   $C2E5
,CC20 60           RTS
,CC21 78           SEI                Cold-start entry
,CC22 A2  FF       LDX   # $FF
,CC24 9A           TXS                Resetta stack
,CC25 A9  E4       LDA   # $E4
,CC27 8D  FA  FF   STA   $FFFA        (FFFA) ─┐
,CC2A 8D  FC  FF   STA   $FFFC        (FFFC)  │ = $E4C2
,CC2D 8D  FE  FF   STA   $FFFE        (FFFE) ─┘
,CC30 A9  C2       LDA   # $C2
,CC32 8D  FB  FF   STA   $FFFB
,CC35 8D  FD  FF   STA   $FFFD
,CC38 8D  FF  FF   STA   $FFFF
,CC3B 20  F5  CB   JSR   $CBF5        Main routine di inizializzazione
,CC3E 20  AE  DB   JSR   $DBAE        Controlla i codici di «GEOS BOOT» e setta (FFFE) = $E2DC
,CC41 20  A3  CD   JSR   $CDA3        Clear screen
,CC4A 20  33  FD   JSR   $FD33        Controllo
,CC47 20  ED  CD   JSR   $CDED
,CC4A A9  01       LDA   # $01
,CC4C 8D  8D  84   STA   $848D        Numero drive
,CC4F A9  00       LDA   # $00
,CC51 8D  65  84   STA   $8465        Default printer-driver name
,CC54 78           SEI                Warm-start entry
,CC55 D8           CLD
,CC56 A2  FF       LDX   # $FF
,CC58 9A           TXS                Resetta stack
,CC59 20  A3  CD   JSR   $CDA3        Clear screen
,CC5C 20  6B  CD   JSR   $CD6B
,CC5F 20  31  CD   JSR   $CD31
,CC62 20  D2  CC   JSR   $CCD2        ($0E) = Nome desktop
,CC65 20  0B  C2   JSR   $C20B        Cerca nome puntato da ($0E) in directory
,CC68 8A           TXA
,CC69 F0  2D       BEQ   $CC98 ──┐    File trovato. File-entry da $8400 a $841D
,CC6B AC  8D  84   LDY   $848D   ▼
,CC6E 88           DEY
,CC6F F0  17       BEQ   $CC88 ──────┐ Finestra file not found
,CC71 A9  09       LDA   # $09 ──┐   │ Prova
,CC73 AE  89  84   LDX   $8489   │   │ altro
,CC76 E0  08       CPX   # $08   │   ▼ device
,CC78 F0  02       BEQ   $CC7C   │
,CC7A A9  08       LDA   # $08   │
,CC7C 20  B0  C2   JSR   $C2B0 ──┘
,CC7F 20  D2  CC   JSR   $CCD2 ──┐     ritenta
,CC82 20  0B  C2   JSR   $C20B ──┘
,CC85 8A           TXA
,CC86 F0  10       BEQ   $CC98 ──────┐ File trovato
,CC88 A9  CC       LDA   # $CC       │
,CC8A 85  03       STA   $03         ▼
,CC8C A9  E4       LDA   # $E4
,CC8E 85  02       STA   $02           (02) = CCE4
,CC90 20  56  C2   JSR   $C256         Visualizza finestra
,CC93 78           SEI               ▲
,CC94 A9  08       LDA   # $08       │ obbligato
,CC96 D0  E4       BNE   $CC7C ──────┘
,CC98 A9  00       LDA   # $00
,CC9A 85  02       STA   $02           indica che i dati del file sono nel Memo-sector
,CC9C 20  D2  CC   JSR   $CCD2         (0E) = nome
,CC9F 20  08  C2   JSR   $C208       ▲ Lo carica
,CCA2 8A           TXA               │
,CCA3 D0  E3       BNE   $CC88 ──────┘ errore
,CCA5 A9  00       LDA   # $00
```

```
,CCA7 85   02        STA  $02
,CCA9 AD   4C   81   LDA  $814C
,CCAC 85   11        STA  $11
,CCAE AD   4B   81   LDA  $814B
,CCB1 85   10        STA  $10
,CCB3 78             SEI
,CCB4 D8             CLD
,CCB5 A2   FF        LDX  # $FF
,CCB7 9A             TXS
,CCB8 20   04   CF   JSR  $CF04
,CCBB 20   6B   CD   JSR  $CD6B
,CCBE 20   31   CD   JSR  $CD31
,CCC1 20   C0   DF   JSR  $DFC0
,CCC4 20   DB   CE   JSR  $CEDB
,CCC7 A6   11        LDX  $11
,CCC9 A5   10        LDA  $10
,CCCB 20   D8   C1   JSR  $C1D8        Lancio del programma
,CCCE 58             CLI
,CCCF 4C   C3   C1   JMP  $C1C3
,CCD2 A9   CC        LDA  # $CC
,CCD4 85   0F        STA  $0F
,CCD6 A9   DB        LDA  # $DB
,CCD8 85   0E        STA  $0E
,CCDA 60             RTS
```

Tavola 10

```
,6000  78            SEI              Disabilita interrupt
,6001  A9  30        LDA  # $30
,6003  85  01        STA  $01         Seleziona tutta RAM
,6005  A9  18        LDA  # $18
,6007  85  02        STA  $02
,6009  A9  60        LDA  # $60       ($02) = $6018 inizio vettore indicizzato
,600B  85  03        STA  $03
,600D  20  73  CE    JSR  $CE73       Rilocamento del vettore
,6010  A9  37        LDA  # $37
,6012  85  01        STA  $01         RAM/ROM di default
,6014  58            CLI              Abilita interrupt
,6015  4C  21  CC    JMP  $CC21       Cede controllo alla cold-start di GEOS
,6018  AE  DB  0B    LDX  $0BDB       Vettore indicizzato
,601B  A9  E2        LDA  # $E2
,601D  8D  FF  FF    STA  $FFFF
,6020  A9  DC        LDA  # $DC
,6022  8D  FE  FF    STA  $FFFE
,6025  60            RTS
,6026  44            ???
,6027  CC  03  EA    CPY  $EA03
,602A  EA            NOP
,602B  EA            NOP
,602C  00            BRK
,602D  00            BRK
```

Tavola 11

```
Finestra 1

0400  20  F5  CB  JSR  $CBF5   Resetta.
                               Abilita controllo mouse. Valida se avete già riesumato GEOS
0403  20  AE  DB  JSR  $DBAE   almeno una volta, in modo che $DBAE setti ($FFFE) = $E20C
0406  20  A3  CD  JSR  $CDA3   Pulisce schermo
0409  A9  04      LDA  # $04
040B  85  03      STA  $03
040D  A9  1B      LDA  # $1B   (02) = 041B inizio vettore
040F  85  02      STA  $02
0411  20  56  C2  JSR  $C256   Disegna e aspetta controllo mouse
0414  4C  54  CC  JMP  $CC54
0417  00          BRK
0418  00          BRK
0419  00          BRK
041A  00          BRK
041B  01  10      ORA  ($10,X)
041D  A0  10      LDY  # $10
```

```
041F  00              BRK
0420  15  01          ORA  $01,X
0422  0B              ???
0423  10  10          BPL  $0435
0425  4E  04  0b      LSR  $0B04
0428  10  25          BPL  $044F
042A  61  04          ADC  ($04,X)
042C  0B              ???
042D  10  40          BPL  $046F
042F  7C              ???
0430  04              ???
0431  0B              ???
0432  20  55  94      JSR  $9455
0435  04              ???
0436  0B              ???
0437  10  70          BPL  $04A9
0439  A7              ???
043A  04              ???
043B  0B              ???
043C  10  85          BPL  $03C3
043E  C8              INY
043F  04              ???
0440  01  19          ORA  ($19,X)
0442  75  02          ADC  $02,X
0444  19  55  03      ORA  $0355,Y
0447  19  35  04      ORA  $0435,Y
044A  19  15  00      ORA  $0015,Y
044D  00              BRK
```

Tavola 11/b

```
:044E 1B C3 48 45 20 56 45 20 .Che ve
:0456 4E 45 20 50 41 52 45 20 ne pare
:045E 44 49 00 51 55 45 53 54 di@quest
:0466 41 20 46 49 4E 45 53 54 a finest
:046E 52 41 20 47 52 41 46 49 ra grafi
:0476 43 41 3F 3F 3F 00 18 CE ca???@.N
:047E 45 4C 20 50 52 4F 53 53 el pross
:0486 49 4D 4F 20 4E 55 4D 45 imo nume
:048E 52 4F 20 44 49 00 1A C3 ro di@.C
:0496 4F 4D 4D 4F 44 4F 52 45 ommodore
:049E 20 C7 41 5A 45 54 54 45  Gazette
:04A6 00 1B 18 56 49 20 53 56 @..vi sv
:04AE 45 4C 45 52 45 4D 4F 20 eleremo
:04B6 49 20 53 45 47 52 45 54 i segret
:04BE 49 20 47 52 41 46 49 43 i grafic
:04C6 49 00 19 50 45 52 20 4F i@.per o
:04CE 54 54 45 4E 45 52 4C 41 ttenerla
:04D6 21 21 21 00 00 00 00 00 !!!@@@@@
```

Tavola 12

```
$8506    Movimento

#$FF     nessun movimento.
#$00     destra
#$01     su destra
#$02     su
#$03     su sinistra
#$04     sinistra
#$05     giu' sinistra
#$06     giu'
#$07     giu' destra
```

Tavola 13

È la routine di clock per temporizzare la trasmissione dei dati realizzata con la scansione del quadro del pannello elettronico. La temporizzazione è:

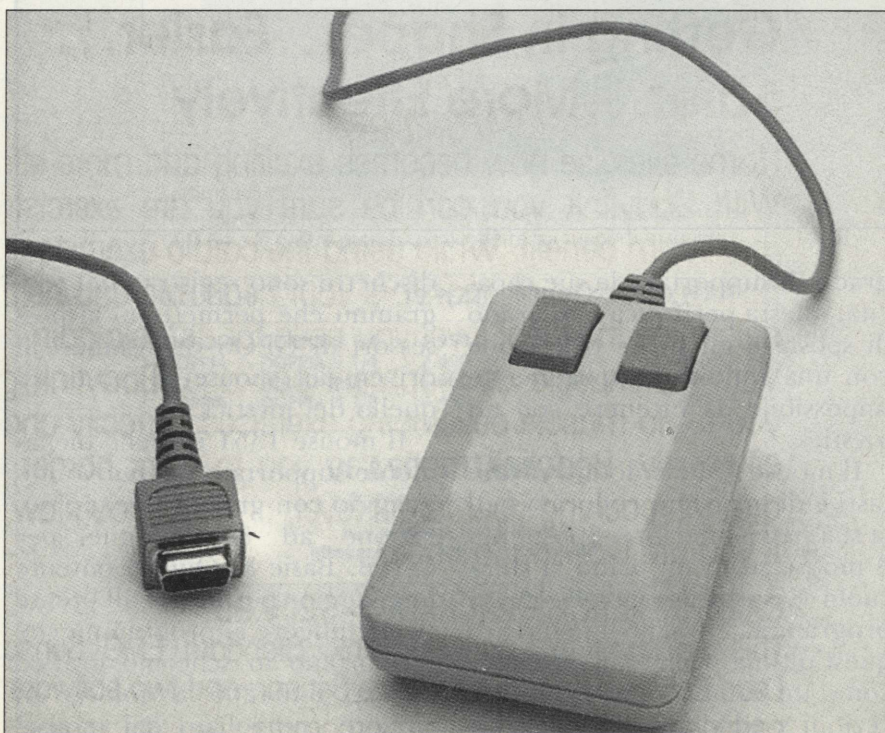
Raster $D012	Stato
00	OK
01	CICLA
02	CICLA
03	OK
04	OK
05	OK
06	OK
07	OK
08	OK
09	CICLA
0A	CICLA
0B	OK
0C	OK
0D	OK
0E	OK
0F	OK

questo da Raster = 00 a Raster = FF

Nuove potenzialità per le macchine ad 8 bit della Commodore

Di Louis R. Wallace

Fino a non molto tempo fa il mercato degli home computer era considerato come un settore morto, dal quale non ci si poteva più aspettare nessun nuovo prodotto o innovazione di rilievo. In un anno le cose sono invece cambiate molto ed anche con l'introduzione del super micro Amiga, la Commodore non ha dimenticato la propria linea di prodotti ad 8 bit ed ha realizzato nuove e potenti periferiche, capaci di espandere le potenzialità di queste macchine molto al di là di quanto ci si sarebbe mai potuto attendere al momento della loro prima introduzione sul mercato. Questi prodotti sono stati presentati al pubblico in occasione dell'importante rassegna statunitense CES (Consumer Electronics Show, n.d.r.) di gen-

naio, che si è tenuta a Las Vegas, in Nevada.

Ho anche avuto la possibilità di utilizzare personalmente alcuni di questi prodotti, come il mouse 1351 ed il drive 1581, per il mio lavoro di giornalista e programmatore, e posso garantire che si tratta di strumenti veramente eccezionali.

Mouse 1351

Una delle più grandi sorprese è stata proprio l'introduzione del mouse 1351 ($49.00, n.d.r.), un vero mouse proporzionale per C-64 e C-128, che funziona esattamente come quello dell'Amiga, del Macintosh e dell'Atari ST. Unito a software in grado di supportare le sue capacità, questa periferica è in grado di spostare gli sprite o le icone con una fluidità semplicemente impossibile da ottenere con un joystick.

Il mouse 1351 è dotato di due tasti e di una sfera collocata nella sua parte inferiore. Spostando il mouse su di un piano la sfera ruota e, sotto il controllo di un programma, il movimento di quest'ultima viene interpretato come un cambiamento nelle direzioni x ed y. Diversamente da un joystick, che dispone di otto possibili direzioni, il mouse 1351 può muoversi in 255 direzioni diverse in modo quasi simile ad un joystick analogico per Apple II. È inoltre possibile farlo lavorare come un semplice joystick, tenendo premuto il tasto destro al momento dell'accensione del computer.

Naturalmente, per poter sfruttare pienamente le possibilità operative di questo nuovo prodotto, è necessario disporre di software che ne supporti le caratteristiche. Il mouse viene venduto corredato di un disco che contiene alcune semplici demo sul movimento di uno sprite a forma di freccetta. Sul retro del dischetto sono registrati dei programmi che permettono ai possessori di GEOS di installare il driver del mouse al posto di quello del joystick.

Il mouse 1351 è anche pienamente supportato dal nuovo linguaggio con grafica in alta risoluzione ad 80 colonne per C-128, Basic 8. Questo potente linguaggio permette all'utente di aggiungere ai programmi che supportano lo schermo grafico bitmap 640x200 a colori, un cursore controllato dal mouse. Un altro package commerciale che supporta il nuovo mouse è la nuova versione ad 80 colonne di Paperback Writer della Digital Solutions.

Espansione RAM 1764

È sicuramente da molto tempo che i possessori di C-64 desiderano più memoria RAM per il loro elaboratore. Proprio per venire incontro a questa richiesta è stato realizzato il modulo di espansione RAM 1764 ($129.00, n.d.r.), una cartuccia di espansione che fornisce 256K RAM aggiuntivi. Il sistema viene venduto insieme ad un nuovo trasformatore che supplisce al maggior bisogno di energia causato dall' aggiunta di questa device. La nuova memoria RAM disponibile non si aggiunge direttamente a quella programmabile, non si deve perciò pensare che sia possibile, con l'acquisto di questa device, far girare programmi da 256K sul 64. Il suo utilizzo è invece quello di una RAM-magazzino al cui interno è possibile registrare programmi e dati cui accedere in qualsiasi momento istantaneamente.

Questa nuova cartridge viene venduta insieme a due programmi che permettono all'utente di sfruttare immediatamente le sue potenzialità. Il primo è un programma «RAM disk» che fa sì che la cartuccia agisca come un disk drive. La più grande differenza rispetto ad un disk drive consiste nei tempi di trasferimento dati che sono incredibilmente più veloci (più di 1000 volte rispetto a quelli di un 1541, n.d.r.) e nel fatto che la memoria è volatile. Ciò significa che, se non si vogliono perdere tutti i dati, prima di spegnere il computer, è necessario trasferire su un floppy il contenuto del «RAM disk».

L'altro programma è una speciale forma di «RAM disk», dedicata agli utenti di GEOS, che consente di aumentarne notevolmente la potenza, riducendo considerevolmente la necessità di questo sistema operativo di ef-

fettuare continui accessi al disk drive.

Non è ancora stato chiarito dalla Commodore se verrà aggiunto al Basic del C-64 qualche nuovo comando che permetta di utilizzare le espansioni RAM in modo analogo a quelle per C-128 (cfr. le funzioni stash, fetch e swap, n.d.r.). Anche se non sarà così, senza dubbio qualche utente riuscirà a trovare un modo per accedere alla memoria aggiuntiva direttamente dal Basic.

Il disk drive 1581

Il 1581 è un disk drive da 3.5" per C-64 e C-128 in grado di registrare più di 800K di dati. Se utilizzato con un C-128 esso risulta più veloce di 15 volte rispetto ad un 1541 e di 2 volte rispetto ad un 1571. Lo stesso dischetto è contenuto in un guscio di plastica rigida che lo protegge molto meglio di quanto si possa fare con un floppy da 5.25". Per diversi mesi mi sono servito del 1581 sul C-128 con il Basic 8, e ne sono rimasto veramente soddisfatto. Oltre alla sua grande velocità e capacità di immagazzinamento, una delle peculiarità di questo drive consiste nella possibilità di servirsi di sub-directory, caratteristica che potenzia notevolmente il sistema di gestione dei file. Il 1581, inoltre, offre un set di comandi DOS molto migliorato, simile a quello dei sistemi in MS/DOS. È possibile, per esempio, ottenere una directory di tutti i file che finiscono col suffisso .dat, utilizzando semplicemente il comando DIRECTORY "* dat".

Non bisogna inoltre dimenticare, tra i motivi d'interesse per questa periferica, l'annunciata compatibilità con 1541 e 1571 (del resto indispensabile, n.d.r.) e con il modo CP/M del C-128.

Disponibilità e prezzi

La commercializzazione nel nostro Paese del mouse 1351, dell'espansione di memoria per C-64 1764, del disk drive 1581, unitamente alle espansioni di memoria per C-128 1700 e 1750, è prevista per il mese di maggio. I prezzi non dovrebbero subire sensibili variazioni rispetto a quelli praticati negli Stati Uniti. ■

Copie bit a bit sul 1541

Alla scoperta del disk drive 1541.

Una delle domande più spesso ripetute in relazione al 1541 riguarda il perché con questo drive non si possano effettuare esatte copie bit a bit di una traccia, visto e considerato che il 1541 è in grado di leggere e scrivere interi settori in una sola volta. Perché non estendere questo processo per poter leggere intere tracce? In apparenza può sembrare un'operazione semplicissima da compiere, ma in realtà è praticamente impossibile: il fatto che questo drive abbia creato un disco non significa affatto che sia in grado di copiarlo.

Incominciamo ad esaminare il processo di lettura dei bit nel 1541. Può sorprendere il fatto che l'unica cosa che il 1541 può individuare direttamente è un bit con valore 1. Un bit a 0 può essere individuato soltanto indirettamente rilevando l'assenza di un bit a 1. Potete pensare a un bit come ad un piccolo magnete con un polo Nord (N) ed un polo Sud (S). Il magnete è sempre allineato parallelamente alla traccia sulla quale si trova, ma può sempre posizionarsi sulla traccia in due modi differenti: NS e SN. Si potrebbe dunque pensare che il 1541 si serva di due diversi modi, uno per i bit a «0» ed uno per quelli a «1», ma ciò non corrisponde assolutamente alla verità. Invece che individuare la direzione attuale del magnete, il 1541 può solo rilevare un cambiamento sulla direzione stessa. In altre parole può soltanto determinare quando un magnete è piazzato in modo diverso da quello precedente, come per esempio NS SN o SN NS. In questo caso il secondo magnete viene interpretato come un bit a «1».

Se due magneti sono piazzati nello stesso modo, come NS NS, formano un singolo magnete lungo il doppio; in questo caso il 1541, una volta entrato in relazione col secondo magnete, non individua alcun cambiamento ed interpreta questa mancanza di una variazione come rappresentante un bit a «0». In che modo deve operare il 1541 per determinare il punto in cui ha inizio il secondo magnete, per poter affermare che in quel preciso punto non si è verificato alcun cambiamento? Tutto si basa sulla temporizzazione delle tracce dalla 1 alla 17; per esempio ad un bit è assegnato un massimo di 3,25 microsecondi (millesimi di secondo o ms). Fate conto che il drive abbia appena letto un bit a «1».

A questo punto è in grado di conoscere esattamente il momento in cui il bit si è presentato, in quanto ha rilevato una mutazione nel campo magnetico. Dopo che il bit è stato oltrepassato, il drive può determinare che dopo 3,25 ms si dovrebbe presentare il bit successivo. Se non rileva alcuna variazione nel campo magnetico entro questo tempo, il drive interpreta il bit successivo come un bit a «0».

Affinché questo preciso schema di temporizzazione funzioni, il dischetto deve ruotare ad una velocità molto precisa: 300 giri al minuto. Ciò corrisponde a cinque giri al secondo, cioè

N. ZONA	SERIE SETTORI	BIT RATE: BIT P. SEC.	BIT-TIME: TIME P. BYTE	BYTE-TIME TIME P. BYTE	SETTORI PER TRACCIA	MAX BYTE PER TRACCIA
1	1-17	307,692	3.25 ms	26ms	21	7692
2	18-25	285,714	3.5	28	19	7142
3	26-30	266,666	3.75	30	18	6666
4	31-35	240,000	4.0	32	17	6250

Tavola 1

200.000 ms per ogni giro. Fate conto che il 1541 abbia appena letto un bit a «1» sulla traccia e che il bit successivo sia di nuovo un bit a «1»: se il dischetto sta ruotando troppo lentamente il bit a «1» potrebbe non presentarsi nel tempo stabilito di 3,25 ms. Il drive probabilmente interpreterebbe erroneamente quel bit come un bit a «0», non avendo rilevato alcun mutamento nel tempo stabilito. Verrebbe così inserito un nuovo bit a «0» nei dati disponibili.

Allo stesso modo considerate il caso in cui il dischetto stia ruotando troppo velocemente e che i successivi 2 bit si trovino rispettivamente a «0» e ad «1». In questo caso il secondo bit a «1» potrebbe presentarsi entro i 3,25 ms facendo sì che il drive non rilevi il precedente bit a «0».

Le variazioni di velocità si presentano sempre indipendentemente dalla precisione con cui questa è stata regolata. A causa della resistenza opposta dal dischetto alla rotazione e di altri fattori, la velocità continua sempre a variare, anche se in modo impercettibile. Per risolvere questo problema sono stati adottati due accorgimenti. Il primo è già stato citato nel corso dell'articolo: il calcolo del tempo è sempre basato sul momento in cui l'ultimo bit a «1» è stato rilevato (in modo che questo sia sempre individuato precisamente). Questa funzione è gestita interamente dall'hardware del drive e non può essere modificata dal software. Se ci sono troppi bit a «0» in una fila, un mutamento della velocità può facilmente invalidare il calcolo dei tempi. Per questo motivo i progettisti del 1541 hanno deciso di non mettere mai più di due «0» in fila sul disco, per avere un buon margine di sicurezza nelle operazioni con il drive. Inoltre i normali byte dei dati, prima di essere scritti sul disco, sono convertiti in una speciale forma chiamata codice GCR. Il codice GCR è stato disegnato accuratamente in modo che non risulti possibile alcuna combinazione di byte GCR con 2 bit a «0» allineati. Questa conversione, nel 1541, è effettuata dal software DOS. Ciò significa che è possibile aggirare questa caratteristica controllando il circuito di scrittura attraverso routine particolari. Si è così in grado di scrivere in fila tutti i bit a «0» che si vogliono, ma si ha anche difficoltà a farli leggere precisamente al 1541 e perciò anche ad eseguire una copia bit a bit.

Risultati ancora peggiori si ottengono mescolando diverse densità di bit su una sola traccia. Abbiamo infatti affermato precedentemente che nelle tracce dalla 1 alla 17 ad ogni bit sono assegnati 3,25 ms; sulle altre tracce il calcolo del tempo è diverso, perché diverse sono le dimensioni delle tracce stesse. In un dischetto del 1541 vi sono 4 zone. Ogni zona è costituita da un gruppo di tracce: rispettivamente le tracce 1-17; 18-24; 25-30 e 31-35. In una zona le tracce operano utilizzando tutte lo stesso calcolo di tempo (densità) e hanno dunque tutte lo stesso numero di settori. I diversi parametri di conteggio del tempo sono riportati nella Tavola 1.

Se ci si serve poi di diverse densità su una stessa traccia, ciò probabilmente è in grado di provocare effetti simili a quelli della variazione di densità. Fate conto di scrivere bit sul disco nella zona a densità 1 (3,25 ms per ogni bit) e poi di saltare di colpo alla zona a densità 4 (4,0 microsecondi) per scrivere ciò che rimane. Immaginate ora di provare a leggere la traccia usando un calcolo di tempo di 3,25 ms. Quando giungiamo al punto in cui la densità è stata cambiata i bit cominciano ad essere rilevati più lentamente ed il drive può, proprio per questo motivo, commettere errori. Se a questo aggiungiamo le inevitabili variazioni di velocità, diviene praticamente impossibile effettuare copie bit a bit.

Uno schema di protezione, comunque, può sempre controllare che i dati sulla traccia siano corretti, a patto che si sia a conoscenza del punto in cui avviene il cambiamento di densità. La densità potrebbe, per esempio, mutare subito dopo che si è presentato un determinato gruppo di bit. Appena il drive individua questo gruppo, la routine di protezione può immediatamente invertire la densità e successivamente attendere un altro gruppo di byte specifico. Se non trova il gruppo che normalmente incontra, il disco non è sicuramente quello originale.

Vi sono poi altri fattori, oltre alle densità miscelate ed alle variazioni di velocità, che possono contribuire a rendere impossibile l'esatta riproduzione di una traccia. Questi fattori, di cui abbiamo già parlato nell'articolo «Le limitazioni tecniche del 1541» nel numero 1/86, includono la limitatezza della RAM del drive, la relativa lentezza del processore (1 MHz) e la mancanza di un sensore per l'«index hole».

In definitiva, quindi, abbiamo dimostrato che, inserendo alcuni opportuni cambiamenti, può essere creato senza particolari problemi un efficace schema di protezione. ■

ESPERIENZE DIDATTICHE A CONFRONTO

Questa rubrica si basa su testimonianze di alunni ed insegnanti, che si interessano alla didattica su computer. I contributi editoriali sono grandemente apprezzati. Inviate eventuali materiali (articoli, foto, disegni, descrizioni di esperienze...) a:

Commodore Gazette
Commodore e didattica
Via Monte Napoleone, 9
20121 Milano

COMPUTER E MUSICA

Le strutture scolastiche italiane che si occupano di musica possono essere raggruppate, in ragione delle finalità e degli obiettivi, in sette gruppi:

1) SCUOLA MATERNA
2) SCUOLA ELEMENTARE
3) SCUOLA MEDIA INFERIORE
4) SCUOLA MEDIA SUPERIORE
5) FACOLTÀ E CATTEDRE UNIVERSITARIE
6) CONSERVATORIO E MEDIE ANNESSE/ORIENT. MUS. E IST. MUS. PAREGG.
7) SCUOLE NON ISTITUZIONALI CIVICHE E PRIVATE

Un chiarimento merita la definizione di «scuole non istituzionali»: nelle quali, infatti, si raggruppano varie realtà, anche molto diversificate fra di loro, da quelle che si limitano a copiare il modello «conservatorio» dal programma alla strutturazione oraria, a quelle più innovative.

Quest'ultime sono accomunate dall'obiettivo di sperimentare e promuovere nuove metodologie e pratiche didattiche, cercando da un lato di colmare lacune secolari e dall'altro di rispondere in modo qualificato alle tendenze più eterogenee della musica.

In seconda analisi questi sette raggruppamenti possono rientrare in una più semplice suddivisione tra strutture che si occupano della formazione di base e strutture che operano specificamente nell'ambito musicale professionale ed amatoriale. Schematicamente ricordiamo:

a) STRUTTURE EDUCATIVE E FORMATIVE DI BASE
la fascia che corrisponde alle scuole dell'obbligo che hanno come obiettivo primario la formazione e l'educazione al linguaggio musicale;

b) STRUTTURE FORMATIVE SPECIALIZZATE NELLA MUSICA
le scuole di musica professionali e amatoriali che si propongono, a livelli diversi, di specializzare l'insegnamento relazionato a strumenti o a discipline teoriche musicali.

Mentre non si può parlare di ingresso generalizzato dell'uso dei computer a scopi didattici in queste strutture, è vero però anche che le strutture di base stanno dimostrando un notevole entusiasmo per i sussidi tecnologici, anche perché coinvolte in un progetto educativo che tiene conto di altre discipline ed interessi e, in generale, della realtà di cui il discente è parte.

Le strutture specializzate che, a rigor di logica, dovrebbero essere quelle maggiormente coinvolte e coinvolgibili in queste innovazioni, rimangono in letargo nell'attesa di riforme e riprogrammazioni da decenni annunciate, ma mai realizzate. Questa generalizzazione non contempla alcune esperienze felici che rimangono «mosche bianche» nella realtà educativa musicale pro-

fessionale e che, in ogni caso, quali eccezioni confermano la regola.

Parlare di informatizzazione dei sistemi educativi in musica, significa principalmente parlare di hardware (in questo caso la scelta è piuttosto limitata) e di software che possiamo definire in base all'uso per cui è finalizzato:

1) software in cui si prevede l'uso da parte di un utente per volta

2) software che può essere utilizzato in gruppo e in classe

3) software che diventa supporto didattico per l'insegnante.

Occorre aggiungere che frequentemente ci si può imbattere in software (Y) utilizzabile in modi diversi e, d'altro canto, che esiste software non dedicato espressamente per l'insegnamento che può essere usato in modo ugualmente proficuo a tale scopo. Almeno altri due fattori intervengono nel determinare la realizzazione di software didattico musicale; in primo luogo il valutare ed il tenere presente il livello di diffusione di prodotti hardware con prestazioni musicali nelle strutture scolastiche e domestiche; inoltre il considerare la possibilità di diffusione e distribuzione del prodotto software.

In questo ambito, l'opera forse più completa e specializzata, realizzata in Italia, è disponibile su Commodore 64, primo computer per diffusione nel nostro Paese, con un vantaggioso rapporto prestazioni/costo a cui si può aggiungere una relativa semplicità di programmazione in linguaggi poco evoluti.

I package dedicati alla didattica musicale, prodotti e realizzati in Italia, trovano la loro culla in una scuola di musica non istituzionale, il CEPAM (Centro Permanente Attività Musicali) di Reggio Emilia. Questo dato può fare riflettere sulla volontà che spinge gli operatori di queste strutture a ricercare e sperimentare metodologie e pratiche nuove che, come in questo caso, vengono usate e applicate nelle strutture istituzionali e nelle scuole dell'obbligo. L'ideatore di questi «libri interattivi» è il direttore del CEPAM, Giuseppe Codeluppi, che da anni si occupa attivamente delle applicazioni didattiche di nuove tecnologie nell'ambito musicale, attraverso l'ideazione di package e l'aggiornamento/formazione degli insegnamenti della scuola dell'obbligo.

Il primo programma è Musico, pubblicato dalla Commodore Italiana nel 1984 e realizzato con Emanuele Iannucelli, un software che ha preso vita nell'ambito dei corsi di computer music, la cui istituzione all'interno del CEPAM risale a quattro anni fa. Musico, package suddiviso in dieci lezioni, partendo dagli elementi alla base del suono, traccia un percorso che tocca gli argomenti fondamentali dell'educazione musicale, cioè gli intervalli, il riconoscimento delle note e la lettura ritmica. Per la sua impostazione è indicato per la scuola dell'obbligo ed è stato progettato per essere utilizzato non solo dal singolo utente, ad esempio nella misurazione e valutazione nella lezione test, ma, e questo vale per la maggioranza delle lezioni, da piccoli gruppi. Le lezioni dedicate alla lettura ritmica permettono infatti di comporre, eseguire, esemplificare ed avere un controllo in tempo reale di ciò che viene scritto direttamente, servendosi della notazione musicale tradizionale.

Il secondo software didattico presente sul mercato italiano è 7 Note Bit, pubblicato dal Gruppo Editoriale Jackson; distribuito in edicola per la prima volta nel 1985, è stato quindi raccolto in tre volumi comprendenti quindici fascicoli con relativo software, adesso anche in versione su floppy disk. Il lavoro è rivolto essenzialmente all'utente singolo ed organizzato come un corso progressivo

Giuseppe Codeluppi (primo a sinistra), autore di software didattico musicale

per l'apprendimento della teoria e della pratica musicale, grazie anche all'ausilio di una piccola tastiera di due ottave da sovrapporre al Commodore 64.

7 Note Bit è suddiviso in varie sezioni presenti in tutti i fascicoli, in particolare ricordiamo:

– argomenti musicali di carattere generale trattati solo su carta (come si parla di musica e cenni sul lessico informatico)

– argomenti musicali non determinanti nella formazione musicale trattati su carta e su software

– il corso di musica vero e proprio realizzato su software con istruzioni ed indicazioni d'uso su carta.

L'aspetto più importante, rela-

tivo alla creazione di questi package, consiste nel fatto che non solo sono frutto dell'opera di persone che da anni operano nell'ambito pedagogico musicale, ma che si connotano anche e soprattutto come prodotti basati su una ricerca e sperimentazione condotta «sul campo» proprio nell'ambito musicale. A conforto di questo depone infatti l'adozione dei package in oggetto da parte di molti insegnanti nell'ambito dell'ordinamento scolastico elementare e medio-inferiore.

Occorre a questo punto aprire un dibattito nelle strutture professionali più refrattarie ad innovazioni e modifiche, per dimostrare che l'uso delle nuove tecnologie può portare solo vantaggi ed accrescere il bagaglio culturale dei futuri musicisti che, nell'ambito dell'evoluzione naturale di ogni forma e genere musicale, si troveranno ad essere sempre più coinvolti in applicazioni e tecnologie digitali.

(a cura di Joseph Powsky)
CEPAM
Villa Cougnet, Via Adua 57
Reggio Emilia
tel. 0522/511990

CALENDARIO

La BSD e l'IRSAE Liguria organizzano una serie di seminari relativi all'utilizzo del software didattico. I seminari sono a numero chiuso, la partecipazione è gratuita; per partecipare è necessario presentare la relativa domanda alla BSD.

PROGRAMMA DEI SEMINARI

L'informatica al servizio degli alunni disabili: il computer come ausilio e sostegno al pensiero e all'apprendimento.

Sperimentazione di software per alunni non udenti.

a cura di: Maria Pia Conte, Giampaolo Chiappini, Gisella De Mola, Emilio Moraglia

data: 23 marzo 1987 – ore 14,30-18.

Le strategie di integrazione scolastica per bambini con danno neuromotorio con l'ausilio della tecnologia elettronica.

a cura di: Giovanni Fronticelli, Paola Sarti

data: 24 marzo 1987 – ore 14,30-18.

Software per la comunicazione e la scrittura nei gravi disabili motori.

a cura di: Psiche Giannoni, Alessandra Schiaffino, Gabriella Veruggio

data: 25 marzo 1987 – ore 14,30-18.

Una esperienza di introduzione all'informatica nella scuola materna ed elementare.

a cura di: Giovanni Belgrano, Clelia Mantelli

data: 1 aprile 1987 – ore 14,30-18.

L'intelligenza artificiale nell'insegnamento: i sistemi tutoriali intelligenti.

a cura di: Danilo Fum, Carlo Tasso

data: 4 maggio 1987 – ore 14,30-18.

Il progetto Seraphim per l'insegnamento della chimica nella scuola media superiore.

a cura di: Giuseppe Innotra, Daniela Perugini

data: 13 maggio 1987 – ore 16,18; 14 maggio 1987 – ore 9-12,30/16-18.

BSD
Via all'Opera Pia, 11
16145 Genova
(010/308883)

La SIEM, Società Italiana per l'Educazione Musicale, organizza un convegno nazionale nei giorni 7-8-9 maggio 1987 presso il Centro Studi della Villa Marigola a Lerici (La Spezia). In questo ambito l'8 maggio, alle ore 15.30, G. Codeluppi (CEPAM di Reggio Emilia, n.d.r.) terrà un seminario sull' **uso del computer e delle moderne tecnologie nella didattica musicale.**

Biblioteca Musica
Via Fiume, 72
La Spezia
(0182/20072)

■

ARRETRATI

COMPLETATE LA VOSTRA COLLEZIONE!

NUMERO 1/86 - Telecomunicazioni: guida all'acquisto di un modem. Il disk drive 1541 ed i suoi limiti tecnici. Corso di programmazione in L.M. su C-64. Il C-128: prova. Il nuovo Commodore Amiga a confronto con IBM PC, AT e Macintosh. Reset per il 1541. Amiga, dove fantasia e realtà si incontrano. Fare musica con il C-128. Convertitore grafico per Commodore 64: listato. Cultura e informatica. Software Gallery: Lode Runner Rescue, The Hitchhiker's Guide to the Galaxy, Jet, Scenery Disks. Software Helpline: The Hitchhiker's Guide to the Galaxy, Zaxxon, Wolfenstein, Star Wars, Ghostbusters, Jumpan, Mindshadow e Tracer Sanction.

NUMERO 2/86 - Jack Tramiel. Corso di programmazione in L.M. su C-64. Telecomunicazioni. Il C-128D. Come operare all'interno del disk drive. Computer e grafica. La Commodore e la didattica. La Rom del C-64. Amiga ed Atari 520ST a confronto. Speciale USA: il CES di Las Vegas, il mondo del III Commodore Show di San Francisco. La mappa di memoria del C-128. Fare musica con il 64: Rondò Veneziano. Archivio programmi: listato per C-64. Software Gallery: Turbo Loading Cartridge, Machine Lightning, Basic Lightning, Uridium. Software Helpline: The Hitchhiker's Guide to the Galaxy, Beyond Castle Wolfenstein, Ghostbusters, Cosmic Balance, Jumpman, Star Trek, Avventura nel computer.

NUMERO 3/86 - Nuovi prodotti Commodore: 64C, 1581 e 1802. Linea Commodore PC IBM compatibili. Gli Ampersand file. Geos. Corso di programmazione in L.M. su C-64. Il SIM HI-FI IVES. Software in CP/M per C-128. Amiga: una nuova era nel mondo del PC. La A-Squared e AmigaLive! Le piante parlano: progetto hardware/software. Amiga: tra sogno e realtà. Espansioni di memoria per il C-128. I disk drive 1570 e 1571 a confronto. Prom ed Eprom. Il suono nel C-128. Jenny: listato per C-128. Eseguiamo le routine del 1541. Suoni e musica con l'AmigaBASIC. Software Gallery: Di-Sector V3.0, Project: Space Station, Matrix 128, 3D Graphics Drawing Board, Textcraft, Game Killer, Nexus, Mach 128, 1571 Clone Machine, VizaStar e VizaWrite, The Final Cartridge. Software Helpline: The Dallas Quest, Questron.

NUMERO 4/86 - Smau '86. CLI: Command Line Interface. La compatibilità IBM per Amiga: Sidecar e Transformer. Schema interno dell'Amiga. Genlock: effetti speciali audio e video con l'Amiga. Amiga days '86. Novità Commodore. Geos ed il suo creatore: intervista in esclusiva con il presidente della Berkeley Softworks. Personalizziamo il sistema operativo del C-64. Csa turbo Amiga. La memoria di massa ed i compact disk. Terzo Commodore Show, Los Angeles. Tempo di Biennale, tempo di Amiga. Protezione scrittura ed interruttore per il numero di device: progetto hardware per C-64. Software per Amiga: descrizione di più di 250 programmi. Grafica 128: listato per C-128. Corso di programmazione in L.M. su C-64. Lista delle routine Kernal del C-128. Software Gallery: Johnny Reb II, War Play, Time Trax, Aegis Impact, The Cataloger, Cartuccia ES-9, Online!, Deluxe Paint. Software Helpline: Borrowed Time.

NUMERO 1/87 - Il Commodore 64 ad Hollywood: l'impiego del 64 nel mondo dello spettacolo. L'Amiga nella realizzazione del serial "Amazing Stories". Amiga e Pontaccio: un Tandem vincente. La politica economica della Commodore International: una panoramica sulle condizioni finanziarie della CBM. Gli Idea Processor: utility per l'organizzazione dei propri pensieri. World of Commodore: servizio speciale sulla IV edizione della nota rassegna canadese. Amiga 1060 Sidecar. La grafica e l'Amiga: un'introduzione alle istruzioni grafiche dell'AmigaBASIC. Grafica avanzata per il C -128: programmazione del chip grafico 8563. Geos: le chiavi del regno. Super Basket NBA: simulazione strategica del gioco del basket per C-64/128. Commodore PC 40 AT: un prodotto IBM compatibile all'insegna della tecnologia avanzata. Hardcopy della pagina grafica in alta risoluzione 640x200 pixel per C-128. Gestione magazzino per Commodore 128. Novità e nuovi prodotti. Routine alternate Kernal-Disco. La stampante MPS 1000. Corso di programmazione in L.M. su C-64. Software Gallery: Tomahawk, Flight Simulator II, Partner 128, Deluxe Paint II, Defender of the crown, Blood'n guts, '43 one year after, Mind Walker , Thai boxing, Skyfox. Software Helpline: Flight Simulator II, View to a kill, Ultima IV.

Ritagliare (si accettano anche fotocopie)
e spedire a: COMMODORE GAZETTE
Via Montenapoleone, 9 - 20121 Milano

Nome e Cognome ____________________

Indirizzo ____________________

Città ____________ **C.a.p.** ____________

Desidero ricevere i seguenti numeri arretrati: ____________

Per un totale di Lire (L. 12.000 ognuno): ____________

☐ **Allego assegno bancario**

☐ **Allego fotocopia della ricevuta del vaglia postale**

Firma ____________________

COMMODORE HELPLINE

LA SOLUZIONE AI VOSTRI PROBLEMI HARD E SOFTWARE

Indirizzate qualsiasi Vostra domanda inerente ai computer Commodore a:

Commodore Gazette
Commodore Helpline
Via Monte Napoleone, 9
20121 Milano

D: Anzitutto i più vivi complimenti per la Vostra rivista che, per quanto cara (6000 lire sono oltre 4$), è qualitativamente pari a «RUN», che credo sia una delle migliori riviste USA del settore.

Apprezzo la Vostra campagna antipirateria, che condivido in pieno. Sono appena rientrato dagli Stati Uniti, dove ho risieduto per oltre due anni per motivi di lavoro. Là avevo un C-64 con drive 1541, stampante e monitor ed ho acquistato nel tempo moltissimi programmi originali, che ho poi portato con me al mio rientro in Italia. Vi posso dire che negli USA i singoli rivenditori di software, né si prestano alla pirateria, né la incoraggiano in alcun modo, come invece mi è sembrato si faccia in Italia. Qui pare normale ricevere offerte di programmi copiati dai singoli rivenditori, per non parlare poi dello scandalo di trovare offerte in vendita nelle edicole riviste che, sia su nastro che su disco, offrono copie di programmi originali USA. Volendo si potrebbero fare nomi!

Ciò che non riesco a credere è che tutto questo traffico si svolga apertamente alla luce del sole! Negli USA perfino i vari programmi di copiatura, come Copy II, Kracker Jax... sono acquistabili per corrispondenza direttamente presso i produttori, ma non sono distribuiti dai rivenditori; inoltre, forse con un po' di ipocrisia, vengono posti in vendita con l'avvertenza «il back-up di un programma originale che hai acquistato è un tuo diritto, ma la pirateria è un reato!».

Rientrato però in Italia con tutti i miei dischi, mi sono trovato di fronte ad un problema, apparentemente insolubile, in merito al quale gradirei conoscere il Vostro avviso. Circa il 10% dei miei programmi originali USA, che colà girava senza problemi, qui non va. Nel mio caso sto parlando di circa una ventina di

programmi su disco, per un valore commerciale, da me regolarmente pagato, di oltre 500 dollari USA. Sembra che di alcuni ci siano copie piratesche in giro. Viva la pirateria allora? Vi sembra giusto che io debba ricomprare legalmente ciò che ho già comprato?

Alberto Ziosi
Bergamo

R: *Per quel che riguarda la differenza di prezzo tra la nostra testata e RUN La invitiamo ad un confronto: provi a contare le pagine di pubblicità e quelle redazionali presenti sul mensile americano e Le confronti con le nostre. Si accorgerà che gli enormi introiti pubblicitari della CW Communications non sono comparabili ai nostri e che una copia di Commodore Gazette contiene più del doppio degli articoli di RUN.*

Il problema da Lei riscontrato nell'uso di programmi originali USA in Italia è dovuto alla differente frequenza di clock che intercorre tra gli elaboratori venduti in Europa e quelli commercializzati negli Stati Uniti (cfr. il non funzionamento del programma Gunship, n.d.r.). Per risolvere il problema dovrebbe essere in grado di intervenire Lei stesso sui codici dei programmi per apportare le modifiche necessarie (operazione piuttosto semplice per chi conosce il L.M., n.d.r.). Nel caso in cui non lo fosse può rivolgersi a qualcuno erudito sul linguaggio macchina del C-64.

D: Sono un ragazzo di 17 anni e frequento una scuola di informatica, mi chiamo Silvestro De Angelis. Anche se questa è la prima volta che acquisto la Vostra rivista sono rimasto molto contento e Vi faccio i miei più sentiti complimenti.

Desidero acquistare un C-128 e vorrei sapere quale DRIVE adattargli. Ho inoltre visto in un negozio della mia città, la tastiera del C-128, anche se era la prima volta che la vedevo, nello stesso giorno in un altro negozio ho visto un'altra tastiera. In un primo momento sembravano uguali, ma poi mi sono accorto che erano diverse tra loro: in una c'erano dei tasti con dei caratteri in più. Vorrei conoscere la differenza che intercorre tra i due modelli.

Silvestro De Angelis
Località non spec.

R: *Il disk drive dedicato al C-128 è il 1571. Nel nostro Paese sono stati commercializzati elaboratori con due diverse tastiere: la tastiera americana e quella americana/italiana. Quest'ultima, oltre ad avere la normale disposizione statunitense dei tasti, possiede sovraimpresse in grigio le vocali accentate italiane e la disposizione della Z, della W e dei numeri normalmente utilizzata per le macchine da scrivere commercializzate nel nostro Paese. La selezione tra le due tastiere avviene tramite l'ausilio del tasto ASCII/CC.*

D: Possiedo da qualche tempo un C-128 affiancato da un disk drive 1570. Il computer lo uso, sia per hobby, sia per lavoro e devo dire che il C-128 è una macchina molto versatile, utile e abbastanza economica. Ora vorrei proporVi alcuni quesiti:
– È possibile aggiungere una seconda testina al drive 1570, potendo disporre quindi di due facce come nel 1571, visto che la basetta interna del drive è uguale a quella di un 1571?
– Esistono programmi per il 128, in modo 128, riguardanti la gestione di un modem? Se sì, dove si possono reperire?

Naturalmente faccio i miei più vivaci complimenti per questa rivista commodorissima che spero non cambi, perché, se così fosse, «i commodoriani perderebbero una parte del loro computer».

Magoga Antonio
Motta di Livenza (TV)

R: *Aggiungere una seconda testina ad un 1570 è teoricamente possibile, ma, per trasformarlo in un 1571, si renderebbe poi necessario modificare anche la scheda madre ed il tutto risulterebbe poco conveniente.*

La Omega Soft (Via Borgo Palazzo 13/B, 24100 Bergamo, tel. 035/239751 - 222801) distribuisce un programma, Terminal 128 (Lit. 80.000), funzionante in modo 128 e scritto interamente in linguaggio macchina, dedicato alla gestione di un modem.

D: Sono Carlo, ho 14 anni, e da pochi giorni possiedo un Amiga. Prima di tutto vorrei complimentarmi con Voi per la magnifica rivista che avete messo in commercio. In secondo luogo, vorrei sapere dove è possibile rintracciare il manuale Basic in italiano dell'Amiga e inoltre se il seguente messaggio di errore che emette il computer (qualche volta), in alcuni programmi come ad esempio Workbench, Graphicraft e Textcraft, dipende dal mal funzionamento della macchina: SOFTWARE FAILURE. PRESS LEFT MOUSE BUTTON TO CONTINUE. GURU MEDITATION 00000003.000027D2.

Prima di emettere il messaggio la macchina sembra che si resetti e la luce che indica il funzionamento della stessa si affievolisce ad intervalli. Quando si schiaccia il pulsante sinistro del mouse il computer richiede l'inserimento nel drive del Workbench.

Carlo Antonelli
Cupramontana (AN)

R: *Quanto da Lei riscontrato è un errore software, che nulla ha a che fare con l'hardware della macchina, dovuto a piccoli errori contenuti nel Workbench, nel Kickstart e talvolta anche nelle prime versioni di software commercializzato da terzi. Utilizzando la V.1.2 del Workbench e del Kickstart, unitamente alle ultime versioni dei programmi già disponibili mesi fa, non si dovrebbe incontrare nessun problema. La versione italiana della manualistica*

dell'Amiga deve essere richiesta alla Commodore Italiana (tel. 02/618321).

D: Sono in possesso di un C-128 e sono interessato all'uso del mouse, soprattutto per creare ed usare del software grafico (ho la passione per la grafica e una buona dimestichezza nel programmare) in modo più veloce e pratico, mediante appunto questa periferica. Sono riuscito per mia fortuna ad acquistare un mouse «AMIGA» per poche decine di migliaia di lire (causa grave incidente capitato all'AMIGA), ma dopo averlo provato, vedendo i soli risultati negativi della periferica, ho pensato che Voi mi avreste potuto aiutare.

È possibile utilizzare il mouse dell'AMIGA con un C-128? Se sì, quali modifiche bisogna effettuare (riguardanti la componentistica) sul mouse affinché diventi compatibile? Se no, che mouse mi consigliereste come atto al mio scopo?

Spero tanto di ricevere una risposta positiva!! ...nel frattempo Vi faccio i miei più sentiti complimenti per la rivista che trovo veramente ottima.

Un utente di C-128
Località non spec.

R: *Il mouse dell'Amiga non è compatibile nè con il C-64, nè con il C-128. Le consigliamo di acquistare il nuovo mouse per C-64/128 Commodore 1351.*

D: Sono un possessore di C-128 e ho intenzione di acquistare un Amiga, ma ho ancora dei dubbi su questo computer.

1) L'Amiga usufruisce degli sprite hardware. Quanti se ne possono utilizzare? Ci sono comandi in AmigaBASIC appositi per essi? Qual'è la loro risoluzione? Quanti colori si possono adoperare in uno sprite? Gli sprite si possono utilizzare in tutti i modi grafici dell'Amiga (sul C-128 sono inutilizzabili in 80 colonne) o esistono limitazioni?

2) Il multitasking è caratteristica vera e propria dell'Amiga o bisogna operare qualche ampliamento?

Cardelli Mario
Garrufo (TE)

R: *1) L'Amiga supporta otto sprite che hanno una larghezza fissa di 16 pixel ed una lunghezza massima virtualmente infinita. Gli sprite dispongono di tre colori più il trasparente ed è possibile fonderne due per ottenerne uno solo dotato di 15 colori più il trasparente. Gli sprite sono utilizzabili in qualsiasi risoluzione e sono supportati dall'AmigaBASIC. 2) Il multitasking è una caratteristica progettuale dell'Amiga per il cui uso non è necessaria nessuna espansione.* ■

FILO DIRETTO CON LA COMMODORE

LA COMMODORE RISPONDE AI SUOI UTENTI

Questa è una rubrica indipendente gestita dalla Commodore Italiana. Le risposte ai quesiti dei lettori sono redatte dai responsabili della CBM che rappresentano esclusivamente le opinioni di detta società le quali possono anche discostarsi da quelle della redazione di Commodore Gazette.

Indirizzate tutta la corrispondenza per questa rubrica a:

Commodore Gazette
Filo diretto con la Commodore
Via Monte Napoleone, 9
20121 Milano

Messaggio della Commodore ai propri utenti

Abbiamo accettato con piacere l'invito di Commodore Gazette ad aprire un filo diretto con i nostri utenti per rispondere ai vari quesiti che essi pongono.

Dobbiamo ammettere che sinora la Commodore non ha brillato per tempestività nelle risposte e probabilmente ha dato un'impressione ai propri utenti di disinteresse verso i loro problemi; ciò era dovuto alla particolare struttura commerciale che la Commodore ha utilizzato, che non le permetteva di avere un contatto diretto con i propri utenti. Per ovviare a questi inconvenienti abbiamo da tempo avviato alcune iniziative che hanno riscosso un buon successo, la più importante delle quali è stata l'apertura di 50 «Commodore Computer Center», centri di cultura informatica presenti su tutto il territorio nazionale; abbiamo inoltre aperto un servizio di Hot Line telefonico funzionante tutti i pomeriggi.

Vogliamo accennare brevemente alle linee strategiche di sviluppo che, già da inizio anno con l'arrivo dell'Ing. Assi, nuovo Amministratore Delegato e Direttore Generale, hanno avuto un preciso obiettivo che si può riassumere in due semplici parole: «CUSTOMER SATISFACTION». La Commodore quindi, a differenza del passato, sta intraprendendo azioni che la porteranno ad adattare la propria struttura organizzativa, di distribuzione, tecnica ed amministrativa in funzione di questo nuovo ed importante obiettivo. Prova di ciò, la disponibilità da parte nostra ad iniziare un dialogo diretto con i nostri utenti, Vi chiediamo solo di avere un po' di pazienza e, perché no, di darci una mano a migliorarci per darVi un servizio degno dei tanti affezionati utenti che hanno riposto fiducia in noi.

D: Sono da qualche anno un utente più o meno fortunato della ditta Commodore e vorrei usufruire dello spazio dedicato dalla Vostra rivista al filo diretto con la Commodore Italiana.

Dopo qualche anno di lavoro e, perché no, di gioco con il glorioso CBM 64 ho deciso di passare ad un sistema leggermente più professionale, o che perlomeno avesse sulla carta delle potenzialità leggermente superiori.

Non potendomi permettere il nuovo e veramente potente AMIGA, e soprattutto per voler salvare tutto il software in mio possesso, ho deciso di optare per un 128D, che ho potuto facilmente acquistare con poca spesa, visto che il rivenditore ha ritirato il mio vecchio sistema C-64. Ora sicuramente mi trovo tra le mani una macchina sicuramente più valida e sofisticata del C-64, ma tutte le possibilità restano di gran lunga inespresse data la scarsità di programmi ad

esso dedicati. Questo vale sia per il modo 128, ma soprattutto per il fatidico modo CP/M. Effettivamente la Commodore sbandiera la possibilità di usufruire dei programmi in modo 64, ma che senso ha non poter sfruttare a pieno le possibilità offerte dagli altri due modi?

In definitiva mi sembra poco corretto che una grande ditta come la Commodore abbia un atteggiamento del genere: la Commodore stessa dovrebbe orientare il mercato e curarsi di rendere reperibile tutto il software indispensabile a rendere il 128 una macchina «viva» e non una clonazione, sicuramente in bello, del vecchio sistema CBM 64. Voglio sperare che la mia voce si unisca ad un coro di voci e che finalmente la Commodore Italiana ci voglia dare la possibilità di non pentirci delle nostre scelte.

Ringrazio «Commodore Gazette» per l'opportunità concessami e con l'occasione voglio porgere i miei complimenti a tutta la redazione per l'ottimo lavoro finora compiuto.

Aspetti Claudio
Verona

R: *Il C-128 non è una macchina morta, anzi è sicuramente uno dei sistemi più moderni che si possano trovare nella fascia di prezzo cui fa riferimento.*

In merito al quesito posto, precisiamo che è disponibile il sistema operativo CP/M, distribuito ufficialmente dalla Commodore Italia. Se non è reperibile nei negozi, è perché non viene ordinato dagli stessi ai nostri distributori.

Stesso discorso vale anche per i pacchetti Superbase 128 e Superscript 128, che sono disponibili con manuali in italiano. Consigliamo di esigere dal Vostro negozio che Vi venga procurato il materiale originale.

D: Vi scrivo dopo aver letto la lettera di Luca Diana riguardo al fatto che, secondo lui, basterebbe essere più informati per reperire software originale. Bene, vorrei chiarire a Luca che probabilmente il software per C-64, ora come ora, si potrebbe reperire forse anche al supermarket, data la sua enorme diffusione, ma è meglio che si dimentichi tutto questo se è vero che intende acquistare un AMIGA. Dei circa cento programmi di cui dispongo non ve ne è uno originale ad eccezione di quelli avuti a corredo del computer. E questo nonostante abbia visitato diversi computer shop e software house (tra l'altro ho ricevuto recentemente il catalogo LAGO e non mi sembra di avervi scorto titoli per AMIGA). Dico questo con cognizione di fatto, perché la mia esperienza va ben oltre l'ambito provinciale, ma ho richiesto cataloghi anche a famose software house che mettono continue inserzioni su riviste del settore.

Secondo punto per cui vi scrivo: visto che intendete stabilire un filo diretto con la COMMODORE ITALIA ecco subito caldo caldo un primo quesito: perché a distanza di tanti mesi dall'uscita dei primi AMIGA non si è provveduto a fornire gli acquirenti di una interfaccia per usare il segnale PAL anziché quello NTSC? Eppure a quel che mi risulta non sono pochi coloro che dispongono di questo sistema ai quali si è assicurato al momento dell'acquisto che si sarebbe provveduto in breve tempo.

Come è possibile che ancor oggi non sia stato tradotto il manuale per il mercato italiano, ben sapendo quale notevole importanza rivesta per un corretto uso di questo computer?

Gianni Santi
Cagli (PS)

R: *Gli Amiga venduti in Italia con standard non PAL europeo sono circa 1000.*

Siamo in ritardo con l'approntamento dell'interfaccia esterna NTSC/PAL, perché il fornitore italiano al quale abbiamo commissionato il progetto ci ha a sua volta ritardato le consegne. Riteniamo però che riusciremo a provvedere alla spedizione di tale apparecchio entro il mese di aprile.

Con l'occasione puntualizziamo che l'apparecchio verrà spedito, dietro rimborso delle sole spese vive di costruzione e spedizione (circa 30.000 lire), solo a coloro che avranno spedito la garanzia ufficiale della Commodore Italiana. Non verrà, per nessun motivo, fornito a coloro che detengono un sistema Amiga non PAL o con garanzia non inviata alla data di acquisto.

Ribadiamo inoltre che i sistemi Amiga, così come tutti i prodotti Commodore, sono dotati di garanzia ufficiale Commodore Italiana, che è l'unico documento valido per usufruire del servizio di riparazione durante il periodo di validità della garanzia stessa. Esigete sempre, quindi, che il prodotto che state acquistando sia dotato della garanzia Commodore Italiana.

D: Sono lettore fin dal primo numero di Commodore Gazette e, tralasciando gli elogi, meritatissimi, che molti altri lettori Vi hanno già fatto e Vi faranno, vorrei solo pregarVi di essere più puntuali in edicola e, soprattutto, di essere presenti in TUTTE le edicole: finora per procurarmi i numeri della rivista ho dovuto mettere in atto una vera e propria caccia al tesoro! Approfitto subito della possibilità offerta da «Filo diretto con la Commodore» per esprimere tutta la mia irritazione per il modo di trattare i clienti da parte della Commodore Italiana. Semplicemente gli utenti di prodotti Commodore, dopo l'acquisto, sono abbandonati a se stessi. Un'ulteriore dimostrazione di ciò mi è stata data dalla... puntualità con la quale non mi è stato inviato GEOS dopo l'acquisto del C-64. Ho inviato la cartolina alla Commodore, a mezzo raccomandata con ricevuta di ritorno, negli ultimi giorni di dicembre '86; il tutto è stato ricevuto dalla suddetta società in data 29/12/86. Fino ad oggi di GEOS nemmeno l'ombra!!

Giovanni Addabbo
Castellaneta (TA)

R: *In merito alla lamentela più che comprensibile del Sig. Addabbo, confermiamo che la nostra struttura sta lavorando intensamente per modificarsi e fornire quindi un servizio migliore ai propri utenti.*

Per quanto concerne GEOS, e qui vogliamo rispondere a tutti coloro che sono in attesa di ricevere questo oggetto tanto desiderato, possiamo confermare che entro il mese di aprile spediremo, a tutti coloro che ci hanno inviato la cartolina acclusa al C-64, il programma con manuale in italiano e, come segno di amicizia, un manuale per il C-64.

D: Ho letto sulla Vostra rivista n. 4 la «Protesta per la Commodore» del Sig. Angelo Arpaia di Ottaviano e sono rimasto molto sorpreso di quanto esposto nella Sua pregiata. Da parte mia, non posso che ringraziare la COMMODORE ITALIANA di Cinisello Balsamo (MI), in quanto a seguito di richieste effettuate per ricevere, prima la guida per l'utente della stampante Commodore MPS-803, e poi per il manuale per l'utente dell'unità a disco 1571, in lingua italiana, sono sempre stato esaudito. Faccio rilevare però che il manuale per il drive 1571 è stato spedito il 2.12.1986, data del timbro postale, come stampe, e mi è arrivato il giorno 29.12.1986 (non penso sia colpa della Commodore).

Dovrei fare una richiesta alla COMMODORE ITALIANA: è una buona iniziativa allegare al nuovo 64C il programma GEOS, ma per gli affezionati clienti, che, prima avevano il glorioso C-64, ed ora hanno acquistato il nuovo 128D, non potrebbe la Commodore Italiana inviare un'offerta per poter acquistare direttamente il programma «GEOS» con relative istruzioni in italiano?

Luigi Rota
Trescore Baln. (BG)

R: *In merito alla richiesta del gentile lettore, confermiamo che GEOS sarà messo in vendita al pubblico. Comunicheremo tempestivamente i riferimenti per poter acquistare tale pacchetto.*

D: Sono un affezionato utente Commodore ed ho deciso di scrivere due righe a «Filo diretto» perché vorrei porre alcune domande:
1) Il fantastico «GEOS» è un prodotto irreperibile sul mercato; dopo lunghe ricerche, ho trovato una copia della prima versione, che definirla spazzatura è limitativo, infatti non funzionano tutti i programmi del menù (tanto per dirne una). Nella rivendita Commodore dove ho acquistato il mio C-128 non hanno idea di come si possano reperire gli originali.
2) Nella presentazione di «GEOS» della Vostra rivista (n.3), si parlò di una versione per il C-128. Perché nessuno sa niente di tale versione? Avete detto che «GEOS» si trovava accluso alla confezione del nuovo 64C. Non è vero, poiché al suo interno c'era fino a qualche mese fa solo una cartolina d'ordine, ed un mio amico che l'ha spedita a luglio 1986 sta ancora aspettando, ormai non più fiducioso e contento di tale politica amministrativa. Chiedo dunque che scopo ha di esistere la Commodore Italiana, se non assicura il minimo indispensabile, riguardante specialmente il software, ai suoi utenti? Anch'io sono contro la pirateria, ma almeno grazie a quest'ultima ho la possibilità di non far marcire il mio computer. In ogni caso, acquistare un programma originale, costa così caro, che solo gli utenti privilegiati possono sperare di acquistarlo.

Alberto Leo
S. Teresa di Riva (ME)

R: *Per quel che riguarda GEOS rimandiamo alle precedenti risposte. La versione GEOS per il C128 non ci risulta che possa essere disponibile per il mercato europeo a breve termine (verrà introdotta sul mercato USA nel mese di giugno, n.d.r.).*

È vero che fino ad alcuni mesi fa nella confezione del 64C veniva acclusa una cartolina utile per ricevere Geos. Da dicembre non l'abbiamo più inserita, perché le promozioni, come sempre, hanno una durata limitata nel tempo.

In merito allo scopo dell'esistenza della Commodore Italiana, voglio ricordarLe che, in termini di mercato, quello che la Commodore Italiana ha fatto (oltre 1 milione di computer venduti) non trova riscontro in nessun'altra nazione europea ad eccezione della Germania. È vero anche che abbiamo avuto problemi di servizio, di disattenzione verso i nostri clienti, ma non è stato certo per scelta, ma dovuto a circostanze a volte incontrollabili. D'altronde se proporzioniamo i problemi al volume di sistemi presenti sul mercato, forse la situazione non è così tragica come alcuni vogliono farla apparire. Le confermiano comunque la nostra massima attenzione ai problemi dei nostri utenti e ne avrà dimostrazione nei mesi a venire.

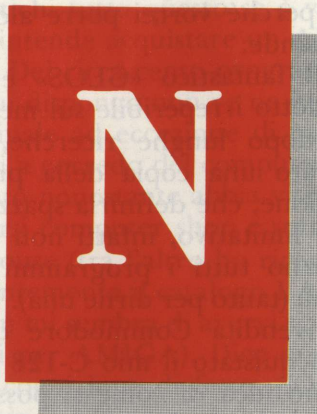
D: Sono un possessore di un AMIGA e vorrei porvi alcune lamentele. Il kit di trasformazione PAL, che dovrebbe essere messo in commercio tra breve, sarebbe giusto che fosse fornito gratuitamente agli utenti, che, come me, hanno comprato l'Amiga all'inizio al prezzo di L. 2.900.000 (ora costa 2.000.000 di lire) e lo stesso vale per la tastiera italiana.

Sarebbe il minimo che la Commodore potrebbe fare per ripagare noi, comuni mortali, che già abbiamo pagato questa macchina 700000 lire in più di chi va ad acquistarla adesso.

Parlando poi dei manuali in dotazione, che sono in inglese, saranno fornite in seguito le traduzioni in italiano?

Andrea Tagliavento
Località non spec.

R: *Per quanto riguarda il problema PAL/NTSC La rimandiamo alla risposta precedente, per il secondo quesito, manuali in italiano sono in arrivo in questi giorni e saranno forniti, dietro rimborso delle sole spese, a coloro che hanno inviato la garanzia ufficiale Commodore Italiana all'atto dell'acquisto.* ■

NOVITÀ HARD E SOFTWARE DALL'ITALIA E DAL MONDO

ITALIA

Nuovo hardware e software per Amiga

La Informatica Italia di Torino distribuisce una serie di pacchetti software/hardware per Amiga. Si tratta di prodotti altamente sofisticati, ma dal prezzo decisamente interessante, che trasformano l'Amiga in un autentico sistema integrato per la produzione di video e per la creazione di musica.

GENLOCK: consente di miscelare immagini create con Amiga con altre provenienti da qualsiasi sorgente video PAL (telecamera, videoregistratore, televisione od un altro Amiga...). Utilizzato in abbinamento a programmi già diffusi sul nostro mercato come Deluxe Paint e Deluxe Video, il Genlock permette di giocare con le immagini in mille modi diversi, dal semplice uso dell'Amiga come titolatrice fino a far muovere oggetti creati dal computer su scenari reali.

DIGIVIEW: è un vero e proprio digitalizzatore video di elevata qualità in grado di produrre immagini in bianco e nero e a colori (32 4096). Collegando una telecamera qualsiasi in bianco e nero ad un Amiga, DigiView consente di trasformare l'immagine ripresa nel suo corrispettivo digitale, che potrà poi essere elaborato con qualunque programma grafico. DigiView può anche essere utilizzato per creare immagini da trasformare in un video, attraverso ad esempio Deluxe Video e Genlock.

SOUNDSCAPE: trasforma l'Amiga in una centralina per la creazione di musica o per l'analisi del suono. Soundscape è un completo programma per campionare e gestire il suono. È fornito con un software particolarmente elaborato il quale permette di effettuare un editing accurato dei segnali campionati e di generare qualunque tipo di strumento per programmi quali Deluxe Music, Instant Music, The Music Studio...

MIDI INTERFACE: questa interfaccia, della stessa azienda produttrice di Soundscape, permette di inviare i suoni campionati ad un impianto al quale siano collegati fino a 16 strumenti via MIDI con uno o più Amiga. È compatibile con tutti i programmi che supportano l'interfaccia MIDI (The Music Studio, Deluxe Music...).

Informatica Italia
C.so Re Umberto, 128
10128 Torino
(011/501647)

Software per Amiga dalla Lago

La Lago ha annunciato l'ormai prossima distribuzione di alcuni dei titoli per Amiga della Mindscape: Defender of the Crown, S.D.I. e Sinbad. È inoltre in preparazione «Lagonews Amiga», un bollettino, inviato

gratuitamente a tutti coloro che ne facciano richiesta, nel quale vengono annunciate periodicamente le novità software disponibili presso la Lago stessa.

Lago s.n.c.
Via Buonarroti, 9
20149 Milano
(02/463659)

Drive esterno da 3.5"

La Niwa distribuisce un disk drive esterno da 3.5" per Amiga il cui prezzo di listino è di Lit. 400.000 (IVA compresa).

Niwa s.n.c.
Via Buozzi, 94
20099 Sesto S. Giovanni
(02/2440776)

ESTERO

Pegasus

La Electronic Arts ha recentemente immesso sul mercato Pegasus ($29.95 + $9 di spese postali per l'Italia), un gioco per gli amanti della guerra navale per C-64.

Electronic Arts Direct Sales
P.O. Box 7530
San Mateo, CA 94403
USA

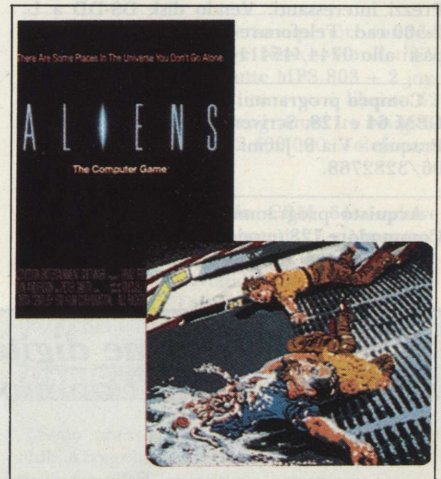
Sub Battle

La Epyx ha realizzato una nuova ed eccitante simulazione per gli amanti della guerra sottomarina: Sub Battle per C-64/128 in modo 64.

Epyx
1043 Kiel C
Sunnyvale, CA 94089
(001/408/7450700) USA

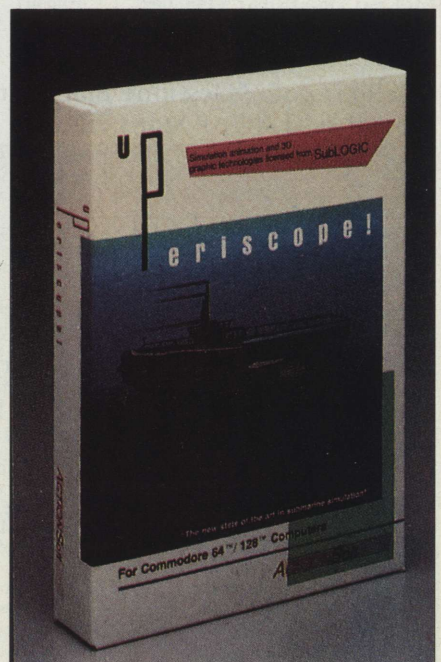
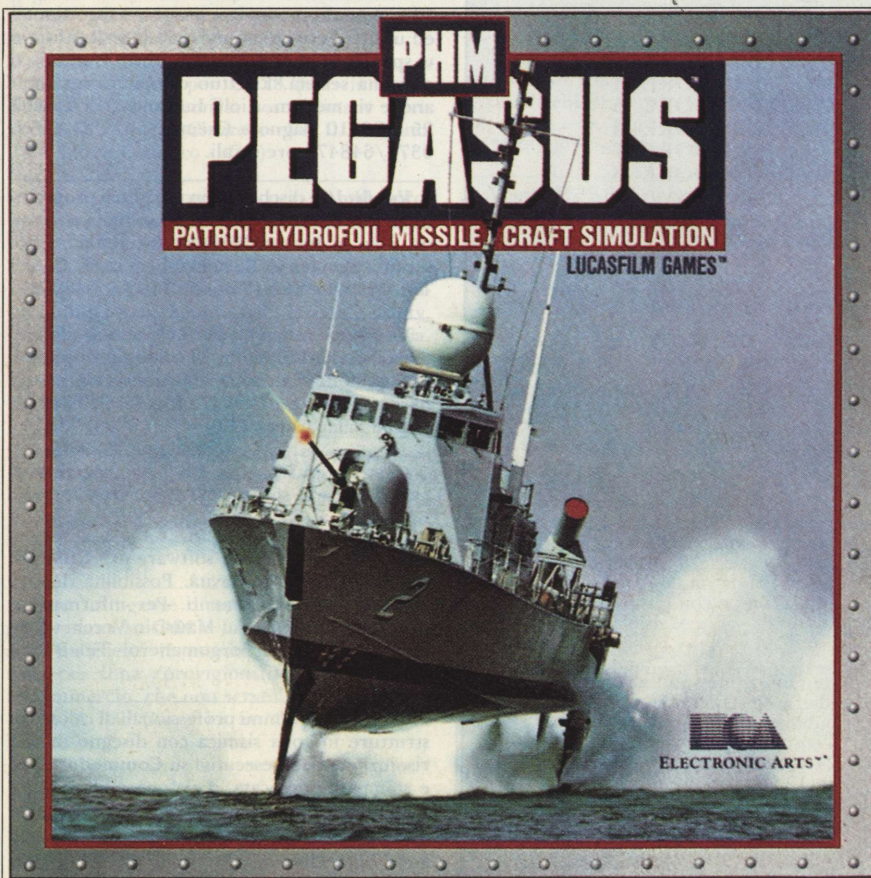
Novità dalla Activision

La Activision ha recentemente presentato: Portal (C-64/128 ed Amiga) ed Aliens (C-64/128).

Activision
2350 Bayshore Frontage Road
Mountain View, CA 94043
(001/415/9600410) USA

ActionSoft

La ActionSoft ha realizzato ThunderChopper (simulatore di volo, $29.95) e Up Periscope (simulatore di guerra sottomarina, $29.95) per C-64/128.

ActionSoft
122-4 S. Race St.
Urbana, IL 61801
(001/217/3671024) USA ■

Classified

SOFTWARE

Vendo programmi per C-128 (modo 128 e CP/M) e programmi per C-64 (tutte le novità). Prezzi interessanti. Vendo disk DS-DD a L. 2.500 cad. Telefonare sabato e domenica, ore pasti allo 0744/454116.

Compro programmi di grafica, disegno per **CBM 64 e 128.** Scrivere o telefonare a Marco Praquin - Via S. Jacini 46 - 00191 Roma - Tel. 06/3282768.

Acquisto programmi di qualsiasi tipo per **Commodore 128** (modo 128 o CP/M). Inviare lista a: Bruno Migliaretti - Via Casaglia 83 - 40135 Bologna.

Vendo programma contabilità ordinaria per **C-64**, prima nota, stampa giornale, stampa schede clienti/fornitori/contabili, stampa bilancio contabile, bilancio di verifica clienti e fornitori. Telefonare Rag. Usuelli 0332/284177.

Vendo le ultimissime novità del software per **C-64** a prezzi eccezionali. Abbonamenti di tutti i tipi per tutte le esigenze a prezzi modici. Cartuccia Freeze Frame a L. 45.000, interfaccia radio per C-64 a L. 35.000. Maurizio Morini - Via Cosenza 122 - 03100 Frosinone - Tel. 0775/200890.

Per **Amiga compro**, cambio software. Inviare liste a: Dalla Valle Roberto - Via Angonese 5 - 36040 Salcedo (VI) - Tel. 0445/888132 (ore pasti).

Per **Amiga, cambio**, vendo solo programmi selezionati. Annuncio sempre valido. Massima serietà. Fabrizio Italia - Via Palestro 101 - 96014 Floridia - (SR).

Compro, cambio, vendo software per **C-64**. Inviatemi la vostra lista. Inoltre cerco programmi videotel e relativi utenti. Scrivere a: Rosario De Siena, Via A. Galante 53 - 80046 S. Giorgio a Cr. - (NA).

Per **C-64 vendo** 3.000 programmi a prezzi fallimentari. Lista gratuita a richiesta. Gianni Mazzesi - Via Cella 329 - 48020 S. Stefano - (RA) - Tel. 0544/573529.

Programmi per **Commodore 64 cambio o vendo** a prezzi stracciati. Telefonare allo 02/591202 o scrivere a: Cannone Antonio - C.so 22 Marzo, 20 - 20135 Milano.

Cambio programmi per **C-64 e 128**. Inviate le vostre liste, rispondo a tutti, se trattasi di cambi. Lovisetto Massimo - Via De Bais 69 - 13062 Candelo - (VC).

Cambio software per **Commodore C-64/128**, in modo particolare programmi gestionali ed utility. Scrivetemi inviando la vostra lista e vi spedirò la mia. Rispondo a tutti. Garantisco massima serietà. Effettuo cambio programmi anche via modem. Aiolfi Luciano - Via Tazzoli 25 - 26010 Bagnolo Cremasco - (CR) - Tel. 0373/648472 ore serali.

Vendo 100 dischetti con i migliori programmi per **C-64/128** al prezzo del solo disco vergine, solo in blocco. Scrivere o telefonare ore serali a: Umberto Ravagnani - Via G. Galilei 10 - Montebello - (VI) - Tel. 0444/649974.

Vendo programma per la gestione di studi tecnici per **C-128** modo 80 colonne. Disponibile il demo. Per informazioni scrivere o telefonare a: Giuseppe Oliveri - Via R. Siciliana 5 - 91020 Salaparuta - (TP) - Tel. 0924/75275.

Cerco utenti **Amiga 1000** per **scambio** di esperienze e di software. Telefonare allo 011/632146.

Vendo programmi e software per **CBM 64**. Disponibili tutte le novità. Possibilità di vantaggiosissimi abbonamenti. Per informazioni telefonare o scrivere a: Maurizio Vecchi - C.so Roma 120 - 28021 Borgomanero - Tel. 0332/844417 (ore pasti).

Vendo programmi professionali di calcolo di strutture in zona sismica con disegno in alta risoluzione degli esecutivi su **Commodore 128** e stampante dedicata. Telefonare allo 0925/28195 Ing. Oliveri Accursio Pippo.

Vendo programmi per **Commodore Amiga**. Ampia disponibilità, prezzi modici. Scrivere o

Come digitare i listati della Commodore Gazette

Simbolo	Premere	Simbolo	Premere
	SHIFT CLR/HOME		SHIFT F1
	CLR/HOME		F3
	SHIFT CRSR ⇆		SHIFT F3
	CRSR ⇆		F5
	SHIFT CRSR ⇅		SHIFT F5
	CRSR ⇅		F7
	CTRL 0		SHIFT F7
	CTRL 1		SHIFT ↑
	CTRL 2		@
	CTRL 3		COMMODORE 1
	CTRL 4		COMMODORE 2
	CTRL 5		COMMODORE 3
	CTRL 6		COMMODORE 4
	CTRL 7		COMMODORE 5
	CTRL 8		COMMODORE 6
	CTRL 9		COMMODORE 7
	F1		COMMODORE 8

ERRATA CORRIGE

Nell'articolo "Gestione magazzino per Commodore 128", apparso sul n. 1/87, è stato rilevato un errore presente nel listato a pag. 94. Sostituire alla linea 610 R$ con P$.

Manoscritti: le collaborazioni in forma di manoscritti, disegni e/o fotografie, sono molto gradite e verranno considerate come possibili per la pubblicazione. La Commodore Gazette non si assume responsabilità per perdite o danni al materiale. Pregasi allegare una busta affrancata ed indirizzata per ogni articolo. Il pagamento per materiale non richiesto viene effettuato solo in seguito all'accettazione da parte della redazione. **I contributi editoriali** (di qualunque forma) non si restituiscono. Tutta la corrispondenza editoriale, richieste di annunci, problemi di sottoscrizione abbonamenti, di diffusione e con gli inserzionisti, deve essere indirizzata a: Commodore Gazette - Uffici Editoriali - Via Monte Napoleone, 9 - 20121 Milano.

La Commodore Gazette è un periodico indipendente non connesso in alcun modo con la Commodore Business Machines e con tutte le sue sussidiarie ed affiliate, compresa la Commodore Italiana S.p.A. La Commodore Gazette viene pubblicata mensilmente dalla IHT Technologies, Via Monte Napoleone 9, 20121 Milano. Il costo degli abbonamenti è il seguente: Italia - 10 numeri L. 58.000, 12 numeri L. 69.000, 24 numeri L. 130.000. Estero - Europa L. 100.000 (10 numeri), Americhe, Asia... L. 160.000 (10 numeri). Nessuna parte di questa pubblicazione può essere in alcun modo riprodotta senza il permesso scritto dell'editore. La redazione si adopera per fornire la massima accuratezza negli articoli e nei listati pubblicati. La Commodore Gazette non si assume responsabilità per eventuali danni dovuti ad errori od omissioni.

telefonare a: Roberto Corbelli - Via Giardini 432 - 41028 Serramazzoni - Tel. 0536/952141.

Per **Amiga, C-128, C-64 vendo** tutti i programmi a prezzi bassissimi. Chiedere catalogo gratuito. Mastrangelo Eliseo - Via Casilina 1641 - 00133 Roma - Tel. 06/6151345 dopo le ore 20.00.

Vendo, cambio programmi **Amiga** e MS-DOS (IBM). Sono interessato inoltre a manuali e programmi di grafica e di telecomunicazione. Inviare propria lista. Luca Zammarchi - Via Massetana 13 - 58022 Follonica - (GR).

Vendo, cambio 600 programmi **Amiga**, ultime novità, manuali, prezzi da "click-off!". Fabio Massimo Angeloni - V.le Tito Livio 200 - 00136 Roma - Tel. 06/348358.

Vendo, cambio moltissimi programmi per **Amiga**, arrivi settimanali, prezzi bassissimi. Bianco Massimo - Via Murri 147 - 40137 Bologna.

Cerco utenti **Amiga** per **scambio** programmi. Cerco inoltre programmi videotel Amiga e Amiga meteosat. Annuncio sempre valido. Roberto Oselladore - Via Passo S. Boldo 35/2 - 30030 Favaro - (VE).

Per **Commodore C-64 e 128 disponibili** oltre 2000 programmi (gestionali, linguaggi, ingegneria ecc.) e 600 manuali di istruzione. Telefonare sera o festivi per lista gratuita: Giacomo - 02/2428315.

Cerco software per **C-128** (solo in modo 128/CPM). Inviatemi la vostra lista e vi risponderò. Alfredo Chizzoni - Via L. Da Vinci 18 - 22020 Gironico - (CO).

Utenti **Amiga** contattasi per **scambio** programmi ed esperienze. Risposta sicura. Marino Crippa - Tel. 0362/930458.

Dispongo per **Commodore 128, 64** ed **Amiga** di 3000 programmi e 400 manuali: utility, giochi, gestionali, radioamatori, ingegneria, ecc. Per informazioni scrivere o telefonare a: Cantelli Massimo - Via Corso 40 - 40041 Altedo - (BO) - Tel. 051/871270.

Amiga scambio programmi (250 al marzo '87). Lista disponibile. Scrivere o telefonare (ore 14-15, 21-22) a: Dr. Franco Zappulla - Via Lombardia 14 - 40139 Bologna - Tel. 051/45667.

Sono un agente di commercio e cerco un **programma su C-64** o 128 che mi gestisca un archivio clienti in maniera: ditta per ditta/ zona per zona /provigioni. Non come quelli in commercio, che non servono! Pietro Stabile - C.so A. De Gasperi trav. 395/32 - 70125 Bari.

Commodore 64, 8.000 programmi disponibili, giochi, utility, musicali, contabilità generale, IVA, ingegneria, computi metrici, legge 373, magazzino, studi legali, stazioni servizio, didattici, c/c, corso inglese, copiatori GEOS V 1.3, edilizia, tot revolution, totip, enalotto, studi medici, agenti rappresentanti, studi dentistici, gestione scolastica, aggiornamenti settimanali. Telefonare allo: 080/681474-807011.

Commodore 128, 300 programmi disponibili, tutti con relativo manuale d'uso: giochi, utility, musicali, grafica, backup, processing, data base, linguaggi, tutto il software esistente in modo CPM, aggiornamenti continui. Telefonare allo: 080/807011-681874.

HARDWARE

Vendo plotter Commodore 1520 nuovo con programma "Basotter" per la gestione in Basic su disco ad un prezzo di L. 160.000. Telefonare a Paolo: 02/2590157.

Compro stampante per Commodore 128 e drive 1571. Sono interessato ad utilities per il C-128. Gioia Claudio - Via L. Caldieri 63 - 80128 Napoli.

Vendo penna ottica per C-64 e Vic 20 + joystick "Deluxe" con autofire + saldatore istantaneo "Elto" 100 W + microfono per ascolto a distanza, il tutto a L. 40.000. Telefonare allo 081/923810.

Vendo modem 300/1200 baud per **CBM 64** ad un costo di L. 150.000. Telefonare allo 0544/463589 e chiedere di Paolo.

Vendo C-64 + floppy disk drive + stampante MPS 801 + modem + speeddos già montato + 100 dischi. In regalo, a chi acquista in blocco, Smagic e Turbo Dos. Telefonare allo 0574/620922 e chiedere di Marco.

Vendo MPS 802 stampante per **C-64** in ottimo stato, completa di kit grafico già montato ad un prezzo di L. 350.000. Telefonare allo 0125/48995.

Vendo stampante **MPS 802** con **Eprom grafica**, come nuova, corredata di 30 programmi e di 6 cartucce nastro a L. 495.000. Massimo Raffaele - Via Crescitelli 6 - 20052 Monza - (MI).

Causa passaggio a sistema superiore **vendesi Commodore 128 + drive 1541** + monitor FV + registratore + stampante MPS 803 + 2 joystick, paddle, penna ottica + vari libri + 62 dischetti con circa 500 programmi tra i migliori esistenti. Il tutto a L. 2.400.000. Telefonare allo 0165/43960.

Occasionissima: **vendo CBM 64** + drive 1541 + monitor 1702 + speed dos già montato + 200 giochi + riviste varie al favoloso prezzo di L. 1.100.000 trattabilissime. Gian Claudio De Matera - Via Taro 25 - 00199 Roma - Tel. 06/852045.

VARIE

Sono aperte le **iscrizioni al Commodore club Afragola**. Il club dispone di circa 3.000 titoli per CBM 64/128. Informazioni: C.C.A. Massimiliano Afiero - C.so Italia iso. 11 - 80021 Afragola - (NA) - Tel. 081/8697713.

È stato **aperto un Commodore News Club** con vasta biblioteca di software per utenti VIC-20/64/128. Cercasi soci. Per informazioni telefonare o scrivere a: Rosa Giancarlo - Via Cherso 1 - 07040 Fertilia - (SS) - Tel. 079/930170 (dopo le ore 20.00).

Amiga Freak's Club. È stato fondato un club totalmente nuovo. Contattiamo Amighi per scambio software e/o notizie. Massima serietà e professionalità. Scrivere a: Mrsek Milko Giuseppe - Via Agro 21 - 25079 Vobarno - (BS) - Tel. 0365/598757.

Vendo enciclopedia Jackson "Software" da rilegare a L. 100.000 e **registratore per C-64** non originale a prezzo da concordare. Solo Parma e provincia. Gian Maria Calzolari - Via Emilia Est 208 - 43100 Parma.

CLASSIFIED DELLA COMMODORE GAZETTE È UN MODO ECONOMICO PER INFORMARE LA PIÙ VASTA UTENZA COMMODORE SUI VOSTRI PRODOTTI O SERVIZI.
GLI ANNUNCI NON A SCOPO DI LUCRO, INVIATI DA PRIVATI, VENGONO PUBBLICATI GRATUITAMENTE (COMPILARE L'APPOSITA SCHEDA DI SERVIZIO LETTORI).

Quote: 15.000 lire per linea, minimo 4 linee. Aggiungere 5000 lire per ogni parola in grassetto o 50.000 lire per l'intero annuncio in grassetto.
Condizioni: pagamento anticipato. Vengono accettati assegni e vaglia postali. Gli assegni devono essere intestati a: IHT TECHNOLOGIES s.r.l.
Forma: gli annunci sono soggetti all'approvazione dell'editore e devono essere scritti a macchina o in modo molto chiaro. Una linea equivale a 40 lettere, spazi tra le parole compresi. Pregasi sottolineare le parole che si intendono scrivere in grassetto.
Informazioni generali: gli inserzionisti devono sempre specificare nome e indirizzo completo. Gli annunci appariranno nel primo numero disponibile dopo il ricevimento.
Data di chiusura: il 10 del mese precedente alla data di copertina (per esempio il numero di giugno chiude il 10 maggio).
Inviare il materiale a:
IHT TECHNOLOGIES
UFFICI PUBBLICITARI
VIA MONTE NAPOLEONE 9
20121 MILANO

Indice degli Inserzionisti

* Questo inserzionista preferisce venir contattato direttamente.
Direzione vendite pubblicità:

IHT TECHNOLOGIES S.R.L.
COMMODORE GAZETTE
UFFICI PUBBLICITARI
VIA MONTE NAPOLEONE 9
20121 MILANO
(02) 794181-799492

Per ricevere complete informazioni dai nostri inserzionisti cerchiare il corrispondente numero del Servizio Lettori sulla scheda dello stesso.

Questo indice è da considerarsi come un servizio addizionale. L'editore non si assume alcuna responsabilità per eventuali errori e/o omissioni.

Indirizzare eventuali lamentele riguardanti gli inserzionisti a:
COMMODORE GAZETTE
UFFICI PUBBLICITARI
ATTN. RELAZIONI
INSERZIONISTI
VIA MONTE NAPOLEONE 9
20121 MILANO

Nessuna responsabilità viene altresì assunta dalla Commodore Gazette per eventuali problemi di qualsiasi natura con gli inserzionisti.

APRILE

NEL PROSSIMO NUMERO

- **GEOS 128** - La nuova versione di GEOS per il Commodore 128.
- **CeBIT '87** - Servizio speciale sulla nota manifestazione tedesca svoltasi ad Hannover.
- **Commodore Show S.F.** - Servizio speciale sull'ultima edizione del Commodore Show di San Francisco.
- **Roma Ufficio '87** - La presentazione italiana dei nuovi prodotti della Commodore.
- **La Commodore Italiana** - Intervista in esclusiva al direttore generale e ai direttori vendite della Commodore Italiana.
- **Corso di programmazione** - La settima parte del nostro corso di programmazione in L.M. per Commodore 64.
- **Dump** - Programma in linguaggio macchina per C-128 che lista su schermo le funzioni definite e tutte le variabili, comprese quelle indicizzate, di un programma in Basic.
- **Archivio dischi** - Listato per Commodore 128.
- **C-64/128 ed Amiga** - Listati e sorprese.

SERVIZIO LETTORI

Questa scheda è valida fino al 30 maggio 1987

A. Quali dei seguenti programmi possedete?
- □ 1. Giochi
- □ 2. Educativi
- □ 3. Finanza familiare
- □ 4. Grafici
- □ 5. Spreadsheet
- □ 7. Database
- □ 8. Utilites
- □ 9. Musicali
- □ 10. Professionali (business)
- □ 11. Telecomunicazioni
- □ 12. Altro (specificare) ____________
- □ ____________

B. Quanto contate di spendere in software nei prossimi 6 mesi?
- □ 1. Nulla
- □ 2. Meno di 100.000
- □ 3. 100.000-200.000
- □ 4. 201.000-400.00
- □ 5. 401.000-1.000.000
- □ 6. Più di 1.000.000

C. Quali periferiche possiede?
- □ 1. Disk Drive
- □ 2. Stampante
- □ 3. Modem
- □ 4. Registratore
- □ 5. Monitor b/n
- □ 6. Monitor a colori
- □ 7. Joystick
- □ 8. Espansione di memoria
- □ 9. Tavoletta Grafica
- □ 10. Light pen
- □ 11. Altro (specificare) ____________
- □ ____________

D. Quanto tempo passate alla settimana con il computer?
- □ 1. 1-3 ore
- □ 2. 3-6 ore
- □ 3- 6-10 ore
- □ 4. 10-15 ore
- □ 5. Più di 15 ore

E. Quante persone leggono la vostra copia della Commodore Gazette?
- □ 1. 1
- □ 2. 2
- □ 3. 3
- □ 4. 4

F. Quali computer utilizzate?
- □ 1. C-64
- □ 2. C-128
- □ 3. C-128 D
- □ 4. Amiga 500
- □ 5. Amiga 1000
- □ 6. Amiga 2000
- □ 7. Altro (specificare) ____________
- □ ____________

G. Quante persone utilizzano il vostro computer?
- □ 1. 1
- □ 2. 2
- □ 3. 3
- □ 4. 4 o più
- □ 5. Nessuno

H. Quali stampanti possedete?
- □ 1. Commodore 1520
- □ 2. Commodore MPS-801
- □ 3. Commodore MPS-802
- □ 4. Commodore MPS-803
- □ 5. Commodore DPS-1101
- □ 6. Okidata
- □ 7. Epson
- □ 8. Star Micronics
- □ 9. Cardco
- □ 10.Tally
- □ 11. Panasonic
- □ 12. Gemini
- □ 13. Seikosha
- □ 14. Toshiba
- □ 15. Altro specificare ____________
- □ ____________

I. Quali sistemi Commodore intendete acquistare nel futuro?
- □ 1. C-64
- □ 2. C 128
- □ 3. C128D
- □ 4. Amiga 500
- □ 5. Amiga 1000
- □ 6. Amiga 2000

J. Indicate il vostro interesse su una scala da 1 a 5 (1 minimo, 5 massimo) per le seguenti applicazioni del computer:
- _____ 1. Business
- _____ 2. Educative
- _____ 3. Finanza familiare
- _____ 4. Giochi
- _____ 5. Telecomunicazioni
- _____ 6. Programmazione
- _____ 7. Grafici
- _____ 8. Musicali
- _____ 9. Word processing
- _____ 10. Database
- _____ 11. Spreadshett
- _____ 12. Altro (specificare)

K. Se non siete un abbonato cerchiate 1.

L. Se volete abbonarvi cerchiate 6 ed allegate l'importo relativo.

M. Suggerimenti ____________________________

Servizio Lettori: Per ricevere maggiori informazioni sui prodotti degli inserzionisti della Commodore Gazette, cerchiate il relativo numero. Troverete i numeri vicino all'annuncio stesso. Completate l'intera scheda, mettetela all'interno di una busta affrancata e speditela all'indirizzo riportato nel retro di questo tagliando. In breve tempo riceverete dettagliate informazioni dagli stessi inserzionisti.

1 6 11 16 21	151 156 161 166 171	301 306 311 316 321	451 456 461 466 471
2 7 12 17 22	152 157 162 167 172	302 307 312 317 322	452 457 462 467 472
3 8 13 18 23	153 158 163 168 173	303 308 313 318 323	453 458 463 468 473
4 9 14 19 24	154 159 164 169 174	304 309 314 319 324	454 459 464 469 474
5 10 15 20 25	155 160 165 170 175	305 310 315 320 325	455 460 465 470 475
26 31 36 41 46	176 181 186 191 196	326 331 336 341 346	476 481 486 491 496
27 32 37 42 47	177 182 187 192 197	327 332 337 342 347	477 482 487 492 497
28 33 38 43 48	178 183 188 193 198	328 333 338 343 348	478 483 488 493 498
29 34 39 44 49	179 184 189 194 199	329 334 339 344 349	479 484 489 494 499
30 35 40 45 50	180 185 190 195 200	330 335 340 345 350	480 485 490 495 500
51 56 61 66 71	201 206 211 216 221	351 356 361 366 371	501 506 511 516 521
52 57 62 67 72	202 207 212 217 222	352 357 362 367 372	502 507 512 517 522
53 58 63 68 73	203 208 213 218 223	353 358 363 368 373	503 508 513 518 523
54 59 64 69 74	204 209 214 219 224	354 359 364 369 374	504 509 514 519 524
55 60 65 70 75	205 210 215 220 225	355 360 365 370 375	505 510 515 520 525
76 81 86 91 96	226 231 236 241 246	376 381 386 391 396	526 531 536 541 546
77 82 87 92 97	227 232 237 242 247	377 382 387 392 397	527 532 537 542 547
78 83 88 93 98	228 233 238 243 248	378 383 388 393 398	528 533 538 543 548
79 84 89 94 99	229 234 239 244 249	379 384 389 394 399	529 534 539 544 549
80 85 90 95 100	230 235 240 245 250	380 385 390 395 400	530 535 540 545 550
101 106 111 116 121	251 256 261 266 271	401 406 411 416 421	551 556 561 566 571
102 107 112 117 122	252 257 262 267 272	402 407 412 417 422	552 557 562 567 572
103 108 113 118 123	253 258 263 268 273	403 408 413 418 423	553 558 563 568 573
104 109 114 119 124	254 259 264 269 274	404 409 414 419 424	554 559 564 569 574
105 110 115 120 125	255 260 265 270 275	405 410 415 420 425	555 560 565 570 575
126 131 136 141 146	276 281 286 291 296	426 431 436 441 446	576 581 586 591 596
127 132 137 142 147	277 282 287 292 297	427 432 437 442 447	577 582 587 592 597
128 133 138 143 148	278 283 288 293 298	428 433 438 443 448	578 583 588 593 598
129 134 139 144 149	279 284 289 294 299	429 434 439 444 449	579 584 589 594 599
130 135 140 145 150	280 285 290 295 300	430 435 440 445 450	580 585 590 595 600

Nome e Cognome ____________________

Indirizzo ____________________

Città ________ Prov. ________ C.a.p. ________

Professione ____________ Età ________

COMMODORE GAZETTE • Marzo 1987

SERVIZIO LETTORI

Questa scheda è valida fino al 30 maggio 1987

A. Quali dei seguenti programmi possedete?
- □ 1. Giochi
- □ 2. Educativi
- □ 3. Finanza familiare
- □ 4. Grafici
- □ 5. Spreadsheet
- □ 7. Database
- □ 8. Utilites
- □ 9. Musicali
- □ 10. Professionali (business)
- □ 11. Telecomunicazioni
- □ 12. Altro (specificare) ____________
- □ ____________

B. Quanto contate di spendere in software nei prossimi 6 mesi?
- □ 1. Nulla
- □ 2. Meno di 100.000
- □ 3. 100.000-200.000
- □ 4. 201.000-400.00
- □ 5. 401.000-1.000.000
- □ 6. Più di 1.000.000

C. Quali periferiche possiede?
- □ 1. Disk Drive
- □ 2. Stampante
- □ 3. Modem
- □ 4. Registratore
- □ 5. Monitor b/n
- □ 6. Monitor a colori
- □ 7. Joystick
- □ 8. Espansione di memoria
- □ 9. Tavoletta Grafica
- □ 10. Light pen
- □ 11. Altro (specificare) ____________
- □ ____________

D. Quanto tempo passate alla settimana con il computer?
- □ 1. 1-3 ore
- □ 2. 3-6 ore
- □ 3. 6-10 ore
- □ 4. 10-15 ore
- □ 5. Più di 15 ore

E. Quante persone leggono la vostra copia della Commodore Gazette?
- □ 1. 1
- □ 2. 2
- □ 3. 3
- □ 4. 4

F. Quali computer utilizzate?
- □ 1. C-64
- □ 2. C-128
- □ 3. C-128 D
- □ 4. Amiga 500
- □ 5. Amiga 1000
- □ 6. Amiga 2000
- □ 7. Altro (specificare) ____________
- □ ____________

G. Quante persone utilizzano il vostro computer?
- □ 1. 1
- □ 2. 2
- □ 3. 3
- □ 4. 4 o più
- □ 5. Nessuno

H. Quali stampanti possedete?
- □ 1. Commodore 1520
- □ 2. Commodore MPS-801
- □ 3. Commodore MPS-802
- □ 4. Commodore MPS-803
- □ 5. Commodore DPS-1101
- □ 6. Okidata
- □ 7. Epson
- □ 8. Star Micronics
- □ 9. Cardco
- □ 10.Tally
- □ 11. Panasonic
- □ 12. Gemini
- □ 13. Seikosha
- □ 14. Toshiba
- □ 15. Altro specificare ____________
- □ ____________

I. Quali sistemi Commodore intendete acquistare nel futuro?
- □ 1. C-64
- □ 2. C 128
- □ 3. C128D
- □ 4. Amiga 500
- □ 5. Amiga 1000
- □ 6. Amiga 2000

J. Indicate il vostro interesse su una scala da 1 a 5 (1 minimo, 5 massimo) per le seguenti applicazioni del computer:
- _____ 1. Business
- _____ 2. Educative
- _____ 3. Finanza familiare
- _____ 4. Giochi
- _____ 5. Telecomunicazioni
- _____ 6. Programmazione
- _____ 7. Grafici
- _____ 8. Musicali
- _____ 9. Word processing
- _____ 10. Database
- _____ 11. Spreadshett
- _____ 12. Altro (specificare)

K. Se non siete un abbonato cerchiate 1.

L. Se volete abbonarvi cerchiate 6 ed allegate l'importo relativo.

M. Suggerimenti ____________________________

Servizio Lettori: Per ricevere maggiori informazioni sui prodotti degli inserzionisti della Commodore Gazzette, cerchiate il relativo numero. Troverete i numeri vicino all'annuncio stesso. Completate l'intera scheda, mettetela all'interno di una busta affrancata e speditela all'indirizzo riportato nel retro di questo tagliando. In breve tempo riceverete dettagliate informazioni dagli stessi inserzionisti.

1 6 11 16 21	151 156 161 166 171	301 306 311 316 321	451 456 461 466 471
2 7 12 17 22	152 157 162 167 172	302 307 312 317 322	452 457 462 467 472
3 8 13 18 23	153 158 163 168 173	303 308 313 318 323	453 458 463 468 473
4 9 14 19 24	154 159 164 169 174	304 309 314 319 324	454 459 464 469 474
5 10 15 20 25	155 160 165 170 175	305 310 315 320 325	455 460 465 470 475
26 31 36 41 46	176 181 186 191 196	326 331 336 341 346	476 481 486 491 496
27 32 37 42 47	177 182 187 192 197	327 332 337 342 347	477 482 487 492 497
28 33 38 43 48	178 183 188 193 198	328 333 338 343 348	478 483 488 493 498
29 34 39 44 49	179 184 189 194 199	329 334 339 344 349	479 484 489 494 499
30 35 40 45 50	180 185 190 195 200	330 335 340 345 350	480 485 490 495 500
51 56 61 66 71	201 206 211 216 221	351 356 361 366 371	501 506 511 516 521
52 57 62 67 72	202 207 212 217 222	352 357 362 367 372	502 507 512 517 522
53 58 63 68 73	203 208 213 218 223	353 358 363 368 373	503 508 513 518 523
54 59 64 69 74	204 209 214 219 224	354 359 364 369 374	504 509 514 519 524
55 60 65 70 75	205 210 215 220 225	355 360 365 370 375	505 510 515 520 525
76 81 86 91 96	226 231 236 241 246	376 381 386 391 396	526 531 536 541 546
77 82 87 92 97	227 232 237 242 247	377 382 387 392 397	527 532 537 542 547
78 83 88 93 98	228 233 238 243 248	378 383 388 393 398	528 533 538 543 548
79 84 89 94 99	229 234 239 244 249	379 384 389 394 399	529 534 539 544 549
80 85 90 95 100	230 235 240 245 250	380 385 390 395 400	530 535 540 545 550
101 106 111 116 121	251 256 261 266 271	401 406 411 416 421	551 556 561 566 571
102 107 112 117 122	252 257 262 267 272	402 407 412 417 422	552 557 562 567 572
103 108 113 118 123	253 258 263 268 273	403 408 413 418 423	553 558 563 568 573
104 109 114 119 124	254 259 264 269 274	404 409 414 419 424	554 559 564 569 574
105 110 115 120 125	255 260 265 270 275	405 410 415 420 425	555 560 565 570 575
126 131 136 141 146	276 281 286 291 296	426 431 436 441 446	576 581 586 591 596
127 132 137 142 147	277 282 287 292 297	427 432 437 442 447	577 582 587 592 597
128 133 138 143 148	278 283 288 293 298	428 433 438 443 448	578 583 588 593 598
129 134 139 144 149	279 284 289 294 299	429 434 439 444 449	579 584 589 594 599
130 135 140 145 150	280 285 290 295 300	430 435 440 445 450	580 585 590 595 600

Nome e Cognome ____________________

Indirizzo ____________________

Città ________ Prov. ________ C.a.p. ________

Professione ____________ Età ________

COMMODORE GAZETTE • Marzo 1987

☐ Si desidero inserire gratuitamente un mio annuncio nella rubrica CLASSIFIED (solo per i privati)

☐ Si desidero inserire un mio annuncio nella rubrica CLASSIFIED. Allego assegno, o fotocopia della ricevuta del vaglia postale, per un totale di L.

Attenzione: perché un annuncio venga accettato è necessario che sia compilato anche il questionario presente sull'altro lato di questo tagliando.

TESTO ______________________________

____

____

Inserire all'interno di una busta affrancata e spedire a:

Commodore Gazette
Servizio Lettori
Via Monte Napoleone 9
20121 Milano

- ✂ - - -

☐ Si desidero inserire gratuitamente un mio annuncio nella rubrica CLASSIFIED (solo per i privati)

☐ Si desidero inserire un mio annuncio nella rubrica CLASSIFIED. Allego assegno, o fotocopia della ricevuta del vaglia postale, per un totale di L.

Attenzione: perché un annuncio venga accettato è necessario che sia compilato anche il questionario presente sull'altro lato di questo tagliando.

TESTO ______________________________

____

____

Inserire all'interno di una busta affrancata e spedire a:

Commodore Gazette
Servizio Lettori
Via Monte Napoleone 9
20121 Milano

La rivista per utenti di C-64/128 ed Amiga

COMMODORE GAZETTE

La redazione della Commodore Gazette cerca CAPO REDATTORE per assunzione a tempo pieno.

È indispensabile una spiccata attitudine allo scrivere in tutte le sue manifestazioni unita ad una perfetta conoscenza della lingua italiana e ad una buona cultura generale.
Referenza indispensabile è anche la buona conoscenza della lingua inglese unita alla capacità di tradurre articoli. NON È RICHIESTA la conoscenza specifica del settore dell'informatica né di linguaggi di programmazione. Si richiedono precedenti esperienze di carattere giornalistico e la eventuale iscrizione all'albo dei giornalisti come professionista o come pubblicista.

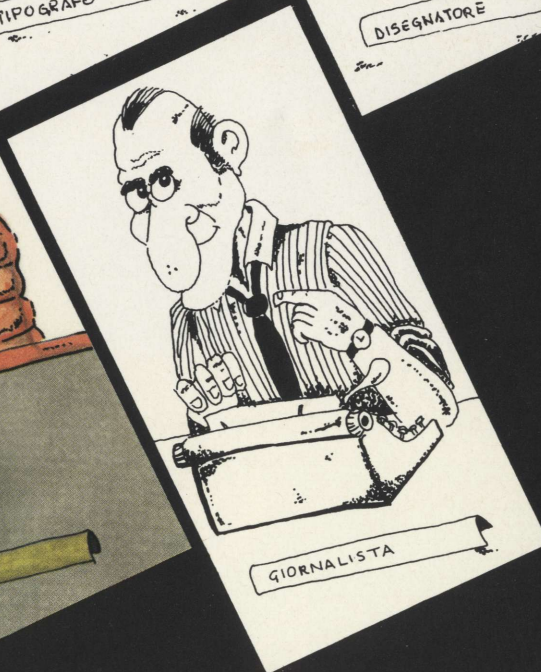
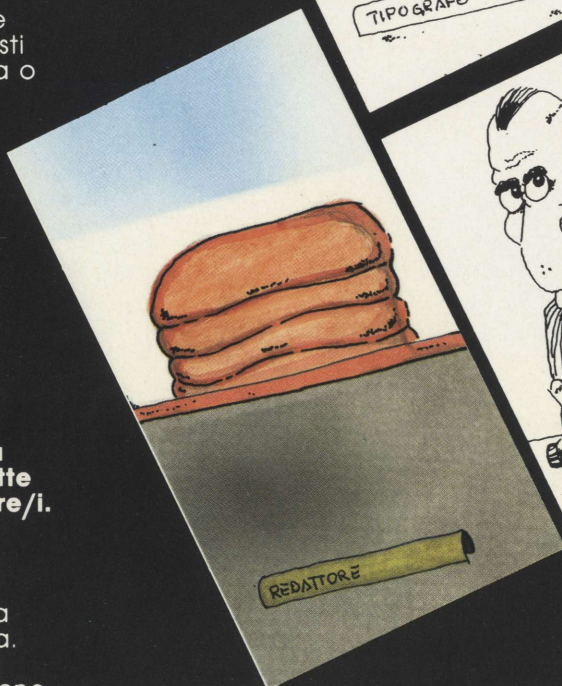
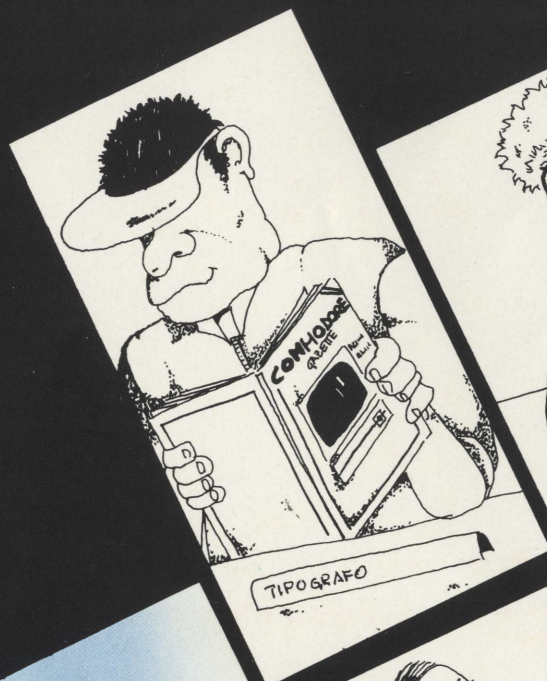
La IHT cerca esperto/i traduttore/i cui affidare la traduzione inglese/italiano di volumi di informatica.

Invitiamo tutti gli interessati ai suddetti annunci a scrivere a:
Commodore Gazette
Ufficio personale
Via Monte Napoleone, 9
20121 Milano

TIPOGRAFO

DISEGNATORE

REDATTORE

GIORNALISTA

La redazione della Commodore Gazette cerca collaboratore/i.

Referenze:

Ottima conoscenza della lingua italiana. Buona conoscenza della programmazione in Basic ed in linguaggio macchina di C-64/128 ed Amiga.

Si garantisce la massima riservatezza.